Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2014

Renato Filipe Ribeiro Lopes

Controlador e device drivers para seguidor solar de 2 eixos

Controller and device drivers for a two axis solar tracker

"We have proved the commercial profit of sun power in the tropics and have more particularly proved that after our stores of oil and coal are exhausted the human race can receive unlimited power from the rays of the sun."

Frank Shuman, New York Times, July 2, 1916

Renato Filipe Ribeiro Lopes

Controlador e device drivers para seguidor solar de 2 eixos

Controller and device drivers for a two axis solar tracker

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Professor Doutor Paulo Bacelar Reis Pedreiras do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro e do Professor Doutor Valter Filipe Miranda Castelão da Silva, Professor Adjunto da Escola Superior de Tecnologia e Gestão de Águeda.

o júri / the jury

presidente / president	**Prof. Doutor José Alberto Gouveia Fonseca** Professor Associado do Departamento de Eletrónica,Telecomunicações e Informática da Universidade de Aveiro
vogais / examiners committee	**Prof. Doutor Paulo José Lopes Machado Portugal** Professor Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto
vogais / examiners committee	**Prof. Doutor Paulo Bacelar Reis Pedreiras** Professor Auxiliar, Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro (orientador)

agradecimentos / acknowledgements

Ao terminar a tese de mestrado não posso deixar de sentir uma certa nostálgia do período que passei na Universidade de Aveiro. É portanto tempo de reconhecer a importância que várias pessoas tiveram ao longo do meu percurso e que me ajudaram a enfrentar este desafio.

Em primeiro lugar agradecer o acompanhamento e ajuda imprescindível bem como a orientao técnica do professor Doutor Paulo Bacelar Reis Pedreiras bem como do professor Doutor Valter Filipe Miranda Castelo da Silva. Pela forma como sempre me trataram depositando a exacta quantidade de confiança para que pudesse ter liberdade na resolução dos problemas que encontrava sem nunca me sentir abandonado por estes.

Aos meus "companheiros" de sala cujo companheirismo permitiu a existência de uma aberta e livre troca de ideias, uma ajuda mútua incansável e a existência de um ambiente informal capaz de criar momentos de verdadeira diversão e de trabalho.

Aos amigos e colegas desta viagem de 5 anos, os inúmeros almoços e jantares, as noitadas de trabalho e o sentimento que existem momentos que irão comigo para o resto da vida.

Por último ao apoio incansável dos meus pais que possibilitou a minha frequência na Universidade de Aveiro, com a quantidade certa de liberdade e responsabilidade, ajudando-me a tornar uma pessoa melhor. À minha namorada que longe sempre esteve perto de mim.

Um muito obrigado a todos!

Palavras-chave Alterações Climáticas, Energia Limpa, Energia Fotovoltaica, Painel Solar, Seguidor Solar

Resumo

Nos últimos anos assistimos ao crescimento da preocupação relativa às alterações climáticas e o seu impacto na vida humana. Assistimos cada vez mais a fenómenos climáticos extremos cujos efeitos são cada vez mais destrutivos e devastadores.

No entanto existe uma maior consciencialização por parte dos consumidores sobre as fontes energéticas da energia que utilizam. Existe também uma maior consciencialização por parte das empresas energéticas sobre a origem das fontes energéticas que providenciam aos seus clientes. Assistimos ainda a nações que até há alguns anos se mostravam extremamente céticas acerca da influência humana nas alterações climáticas a criarem programas de redução das emissões de CO_2 para a atmosfera, nomeadamente os Estados Unidos da América.

Com este objectivo o investimento em energias limpas/renováveis aumentou consideravelmente, tanto na investigação como na criação de grandes parques solares e/ou éolicos. No entanto, o preço proibitivo dos painéis fotovoltaicos ou dos seguidores solares impede os consumidores de conseguirem um retorno do investimento num tempo aceitável, fazendo com que a adesão destes não seja tão grande como poderia ser possível e desejável. É com o objectivo de reduzir o custo associado aos paineis fotovoltaícos e a sua consequente democratização que esta dissertação de mestrado foi executada, aliando o maior rendimento dado pelo seguidores solares de dois eixos a um novo método de controlo livre de soluções proprietárias que não acrescentam valor e apenas oneram os consumidores.

Keywords Global Warming, Clean Energy, Photovoltaic Energy, Solar Panel, Solar Tracker

Abstract

In the last few years we have witnessed the growth in the awareness relative to the effects of the climate change and global warming in the human life. The effects of extreme weather phenomenons are becoming more devastating.

However we have also seen an increase in the consumer awareness relatively to the sources of the energy that they use. There is also an increase in the awareness from the power suppliers about the energy sources that they provide to their costumers. Nations that some years ago stood almost alone in defiance of the Kyoto protocol are now implementing programs that aim the reduction of carbon dioxide emissions to the atmosphere, namely the United States of America.

With this objective, the investment in cleaner energy sources has increased considerably, both in research and in the creation of huge solar/eolian parks capable of producing large amounts of energy. However the prohibitive prices of the solar panels or of the solar trackers impedes the consumer of getting the return of the investment in a satisfactory time reducing the adherence to this kind of solutions. With this problem in our minds, the objective of this master dissertation is to aim for a reduction in the associated cost of the solar tracker solution by implementing new hardware and software capable of performing satisfactorily while being cheaper than the commercial solutions available. This aims to ally the bigger efficiency of the solar tracker solution (two axis) with a new control method free of proprietary solutions that do not increase value, only burden the consumer and thus halting the adoption of this kind of technology.

Contents

References **57**

List of Figures

List of Tables

Acronyms

AADAT Azimuth Altitude Dual Axis Tracker. 7

AC Alternating Current. 10, 11, 17, 48, 49

ADC Analogue to Digital Converter. 49

API Application Programming Interface. 37

CMake CNU Cross Compiler. 33

DC Direct Current. 10, 11, 39, 47, 48

GPIO General Purpose Input/Output. 21–23

HMNAO HM Nautical Almanac Office. 8

HSAT Horizontal Single Axis Tracker. 6

LDR Light-Dependent Resistor. 11, 17, 35

MICA Multiyear Interactive Computer Almanac. 8

MySQL My Structured Query Language. 33, 41, 48, 52, 53

NREL National Renewable Energy Laboratory. 9

NVRAM Non-Volatile Random Access Memory. 37, 38

PCB Printed Circuit Board. 50

PLC Programmable Logic Controller. 2, 10, 15–17, 55, 56

PSA Plataforma Solar de Almeria. 8, 15

RAM Random Access Memory. 12, 29, 41

RTC Real Time Clock. 12, 36, 38, 41, 49, 50

SPI Serial Peripheral Interface. 21

SSH Secure Shell. 48

TSAT Tilted Single Axis Tracker. 6

TTDAT Tip Tilt Dual Axis Tracker. 6, 7

USNO United States Naval Observatory. 8

UT Universal Time. 9

VFD Variable Frequency Drive. iii, 11, 17, 18, 20, 24, 39, 40, 49

VSAT Vertical Single Axis Tracker. 6

Chapter 1

Introduction

1.1 Historical Reference

Frank Shuman was an American engineer and inventor born in 1862. During the course of his life he invented and patented numerous inventions namely the safety glass, the danger signal in railroad crossings and the use of liquid oxygen to propel submarines. These inventions award him recognition and enough money to pursue his dreams. One of those dreams was the use of solar energy to power different kinds of engines. In the beginning of the 20^{th} century he demonstrated the feasibility of the sun's energy to power a small steam toy. Following his discoveries in this field, Frank Shuman formed the Sun Power Company with the intent of building solar powered power plants. In the year 1912, Shuman and his company built the first solar thermal power station in Maadi, Egypt. It was capable of producing 60 to 70 horsepower and delivering almost 23000 liters of water per minute from the Nile river to the surrounding cotton fields thus proving the viability of this kind of solution. With the beginning of the first world war the power station was dismantled to scrap metal to produce weapons for the war thus ending Frank's solar dream. The discovery of new oil deposits and the pressure to extract more fossil fuels due to the war halted any development or commercial interest in the next 60 years (Gornall, 2011).

1.2 Introduction

In recent years the wake of extreme climate changes led to the pursuit for cleaner energy sources (C.B. et al., 2014). From the wide range of environmental friendly sources, the solar energy seems to be the preferred by enterprises and consumers all over the world, especially in places where its availability is paramount. Although the solar energy is widely used, in the form of photo voltaic panels, the technology it uses is fairly inefficient resulting in low energy production and long investment payback periods (McGehee, 2014) .

To resolve, partially, this problem we've seen a rise in the use of solar trackers to boost the energy output by following the sun's position during the day, by maintaining the solar panel in its optimum position for a longer period of time. However this kind of solutions are, usually, proprietary and commercialized at prohibited rates/prices and with little room to user customization. This kind of practice impedes an higher adoption by the consumer maintaining this technology at enterprise level.

We have seen also an increase in the amount of money spent in researching more effective

solar cells and also in lowering their production costs. Some projected solar cells yield 50 % more power than the previous iteration of this technology (Green, Emery, Hishikawa, Warta, & Dunlop, 2012).

The majority of the solutions implemented for controlling the solar panels in the field resort to Programmable Logic Controllers (PLC) which offer some advantages when dealing with systems placed in harsh environments but lack some advanced features. Furthermore they usually are closed-source/proprietary systems leaving consumers and enterprises effectively locked to the manufacturer for firmware updates and new capabilities. They also tend to be pricier considering the basic functionality that they offer.

With the growth of the clean/renewable energy market there is the need to produce cheaper controllers while providing new features that enable to reduce the upkeep costs associated with this kind of systems (EPIA, 2014).

1.3 Motivation

Considering the current status-quo in the cleaner/renewable energy market it is fundamental to create cheaper solutions that offer the same or even more functionalities that the current options in the market offer. For all the pros that a PLC solution offer I consider it to be functionality limited considering the recent boom in affordable and powerful credit-card sized single-board computers, like the Raspberry Pi, which are able to run Linux-like operating systems enabling the use of more advanced functionalities like built-in Ethernet communication for Internet capability while maintaining the existence of input/output ports for controlling devices.

By using this kind of devices as the main controller of a solar tracker we are enabling new kind of functionalities that could help to lower the upkeep costs by allowing remote monitoring of the solar tracker hardware and software preventing unnecessary costs in technician travels and work. And in the event of a technician maintenance visit to the solar tracker, this kind of solution would not require proprietary software, being as easy as connect an Ethernet cable to the controller and type the IP address to access a web interface that gives all the required information about the current status of the system and if is required any kind of action.

However, since the Raspberry Pi, as other similar products, lacks the hardware required to control a solar panel, it is necessary to create a separate control board dedicated to interface the hardware of the solar panel to the Raspberry Pi. Since the hardware present in a solar panel varies with the manufacturer, some attention must be taken to ensure that the control board created can interface with the majority of the configuration existent in the market (*vide* Section 2.2.4).

Finally, by creating an open-source solution we are enabling the user to create their own additions to the platform and with each addition create a better product and faster implementations of the needs of the consumer.

1.4 Thesis Organization

In the Chapter 1 of this dissertation, an introduction is made about the current state of the technology and the motivations behind this dissertation.

In the following chapter, a basic understanding of the wide range of components used to control a solar panel is given to the reader, which serves as basis of understanding of

the following chapters and the chosen architecture. A deeper understanding of the concepts behind the kind of tracking that will be used throughout this dissertation will also be given.

An overview of the market in order to specify the requirements that the create Architecture must meet will be given in Chapter 3.

In the Chapter 4 will be presented the hardware modules created within the chosen architecture. It will be explained why the particular solution was chosen and how it was implemented while providing the necessary schematics and calculations that support the choice.

The same procedure will be made in Chapter 5, where the implemented device drivers will be explained in detail, listing the used programming languages, providing the execution diagrams when needed and showing the functionalities developed.

The tests made to ascertain the capability of the architecture proposed and their results are explained in Chapter 6.

In the last chapter, 7, some brief remarks are made about the feasibility of the solution developed and how it compares against the commercial products existent in the market. An overall evaluation of the controller created will be done and some ways to improve it will also be given.

Chapter 2

State of Art

The purpose of this chapter is to acquaint the reader with some of the technologies used in the field that serve as basis of this master dissertation.

2.1 Solar Panels

A solar panel (*vide* Figure 2.1) is a combination of solar cells electrically connected to increase the overall power output of the system. Each solar cell, or photovaltaic cell, converts the energy from sunlight directly to electricity via the photovoltaic effect. Its characteristics, voltage, current or resistance vary according to the amount of light received.

The efficiency of this conversion is around 21.5 % in the latest commercial products although recent technologies achieve a projected efficiency of 50% (McGehee, 2014, p. 5).

The more common solar cells are made from either crystalline silicon or thin-film, being the former more common than the latter due to it higher conversion efficiency, although being more bulky. Among crystalline silicon based technologies the more prominent are mono-crystalline and polycrystalline cells where the mono-crystalline are more efficient but also more expensive to produce while the polycrystalline are less efficient but also less expensive due to the manufacturing process (Green et al., 2012).

Figure 2.1: Example of an array of solar panels mounted on the roof

2.2 Solar Tracker

The amount of energy produced by a N solar cell panel depends, among other factors, of the angle between the sun's position and the solar panel perpendicular. A solar tracker is a

device characterized by its ability to move the solar panel in order to track the sun's position during the day, thus increasing the energy produced in relation to a solar panel mounted in a fixed structure. However, this kind of equipment requires a bigger initial investment and upkeep costs, since they require active components such as motors and controllers to move the structure that supports the solar panels (Mousazadeh et al., 2009).

The solar trackers can be classified as **one** or **two** axis depending on the number of axis able to move and can also be classified as **passive**, **active** or **chronological**, depending on the type of solar tracking used.

2.2.1 One Axis Solar Tracker

One axis solar trackers are characterized by its ability to track the sun's position along a single moving axis, either Vertical Single Axis Tracker (VSAT), where the azimuth of the panel is fixed and the tracking occurs only in elevation, Horizontal Single Axis Tracker (HSAT) (*i.e.* Figure 2.2), where the elevation of the solar tracker is fixed and the tracking is done in the azimuth angle and the Tilted Single Axis Tracker (TSAT) that allows the movement along a single tilted axis managing to track the sun position along a limited range of elevation and azimuth angles (Wikipedia, 2014).

Figure 2.2: Example of a Horizontal One Axis Solar Tracker

There are some considerations that must be taken into account to choose the single axis tracker to use. Both VSAT and TSAT have the possibility of casting a shadow to nearby solar trackers when used in large field layouts due to their size. The HSAT normally doesn't present this problem and its the most used where a dense concentration of single axis solar trackers are needed. As they have less moving parts than the two axis solar tracker, they are cheaper to maintain and also cost less to install.

If they are of the active type (*vide* Section 2.2.3), some of the energy produced by the solar tracker must be used to power the motors and controller, resulting in a loss of efficiency depending on the power required. Nonetheless they still represent an energy output increase by an estimated amount of 20 to 35 % in relation to an fixed solar panel in the same conditions (Mousazadeh et al., 2009, p. 1814).

2.2.2 Two Axis Solar Tracker

Two axis solar trackers are able to track the sun's position along two moving axis, either being a Tip-Tilted Dual Axis Tracker (TTDAT), where the panel array is mounted on the

top of a single pole which moves in a wide range of elevation and azimuth angles and has low structure footprint, or a Azimuth-Altitude Dual Axis Tracker (AADAT) (i.e. Figure 2.3),where the panel array is mounted along a primary axis, normally the azimuth axis that moves along a ring structure, as is visible in the Figure 2.3.

Figure 2.3: Example of a Two Axis Solar Tracker

These types of solar trackers also present a challenge when planning a large field layout. Their size increases the possibility of casting a shadow to nearby solar trackers, resulting in loss of efficiency. For these types of solar trackers, the TTDAT is significantly more used in large solar tracker power plants because they allow a larger density, given that the AADAT only allows the trackers to be as close as the diameter of the ring.

As it happens with the one axis solar tracker, if they are of the active tracking type, they require power to operate, but in this case, as the tracking is done in two axis, the power required is significantly larger as are the maintenance costs given the number of moving components required. Nonetheless, they normally represent an increase in energy output up to 40% in relation to the fixed solution (Mousazadeh et al., 2009, p. 1814). This increase in performance explains why this is the most used solution in our country.

2.2.3 Passive vs. Active and Chronological tracking

The solar trackers can be divided in three categories:

- Passive
- Active
- Chronological

Passive Tracking

The more common passive trackers use a type of fluid/gel, normally Freon, that when heated by the sun, forces the structure that supports the solar panel to align with the sun position. Further iterations of this technology also implement zones where the sunlight is allowed to pass through the structure and hit it from behind thus further increasing the sensitivity of this system. This kind of system depends heavily on the stability of the fluid/gel properties.

These type of trackers are comparable in terms of performance to the active solar trackers while requiring less initial investment and maintenance costs. The downside of these type of trackers is that they usually work in low efficiency and do not work properly at low temperatures (Mousazadeh et al., 2009, p. 1806).

Active Tracking

The more common active trackers rely on either photo-diodes or chronological data to actively track the sun's positions. This fact requires the usage a controller to monitor or calculate the sun position and act accordingly. Adding this controller not only adds complexity to the system but also increases the costs of the whole system (both the initial investment and the maintenance costs). The photo-diode system reads the luminosity of two or more photo-diode sensors to calculate the sun's position in relation to the solar panel and to properly align the angle of the solar panel in which the energy output is maximum. The chronological system calculates the sun's position resorting to an algorithm which calculates the position according with the solar panel location (Latitude and Longitude) and the current system time. While the photo-diode system is heavily dependent of the calibration of the sensors, the chronological system is prone to error dependent on the limitations of the system that calculates the sun position and due to the inaccuracy of the algorithm used (Mousazadeh et al., 2009, p. 1806-1813).

The active trackers have a greater precision in relation to the passive type. This fact results in a potential increase of performance and the temperature range of operation is only limited by the temperature range of the controller used.

Chronological Tracking

The chronological tracking resides on the principle that the sun position can be calculated, within a certain degree of certainty, by an algorithm. These algorithms, with varying complexities, usually require the geographical location of the solar tracker and the current time and date.

The sun's position relative to an observer, figure 2.4, can be defined by two coordinates:

- Azimuth - coordinate position relative to the south pole (in the northern hemisphere) - (horizontal axis).
- Zenith - coordinate position relative to the normal of the earth (perpendicular) - (vertical axis).

The solar tracking algorithm must be able to deal with the changes in the solar elliptic between seasons, figure 2.5, with the changes of the earth's orbit and with the existence of the gap days.

When using a system with a low concentration of solar panels such as the solar tracker used, an accuracy of 1° because the losses due to misalignment are negligible, *circa* 0.015% (Wikipedia, 2014). Therefore, with this accuracy in mind, there are several ways to calculate the sun position. There is a look-up table based on **The Astronomical Almanac** maintained by the United States Naval Observatory (USNO) and the HM Nautical Almanac Office (HMNAO) and a computer implementation named Multiyear Interactive Computer Almanac (MICA), also provided by the USNO. Another implementation is the PSA algorithm from the Plataforma Solar the Almera, which implementation is open-source and available at the

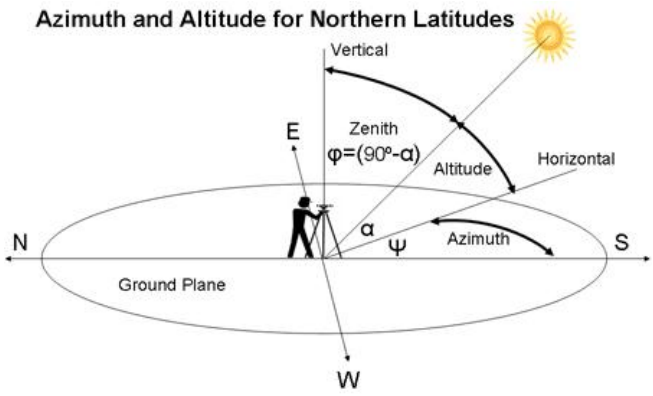


Figure 2.4: Sun position relative to an observer. Source: http://www.mpoweruk.com/

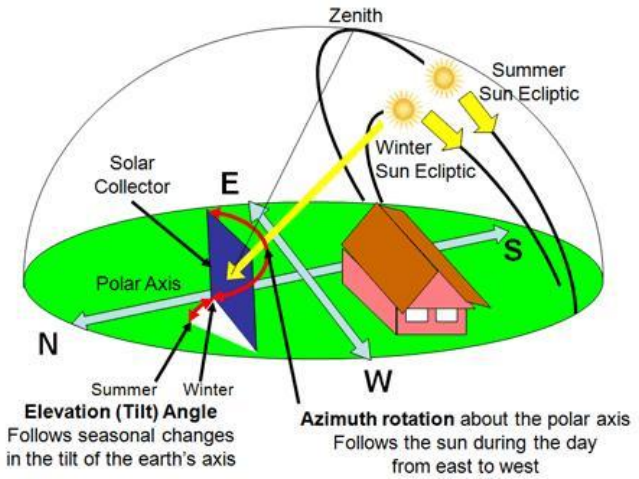


Figure 2.5: Changes in the sun's elliptic between seasons.// Source: http://www.mpoweruk.com/

National Renewable Energy Laboratory (NREL) web page. This algorithm uses the UT to remove the uncertainty caused by the local times zones and can reliably calculate the sun position, recurring to the Julian Calendar and applying the necessary corrections (Reda & Andreas, 2008). Further explanation can be found at the NREL web page.

2.2.4 Common Core Components

Although the main components vary from system to system, the following ones are the most common and will be explained to serve as an introduction so that the next chapters can be easily comprehended as they are used in the controller projected in this dissertation.

- Main Controller
- Linear Actuator
- Variator

- Photo-diode or luminosity sensor
- Anenometer
- Real Time Clock

Main Controller

The main controller of either an active or passive system is the component responsible for the estimation of the sun's position. For this dissertation we will focus in active tracking systems where the most used main controller is the PLC (*vide* Figure 2.6). The PLCs offer robustness and reliability that are fundamental when designing an outdoor system that is subject to harsh conditions. They also offer a simple programming environment, mainly Ladder or Function Block Diagrams, being ideal in system with low complexity while maintaining low upkeep costs and time. The isolation of its input/output ports is ideal in noisy environments, reason why it was widely adopted by the industry. Despite all pros, PLC's usually are proprietary involving the use of proprietary programs and systems with closed-source hardware and software. This fact hinders any customization/upgrade by the consumer and effectively locks the consumer to the schedule/plans of the company that manufactures the system. Some systems use micro-controllers that implement some basic functionalities but these systems are only now appearing in the market (Mikkor & Roosmolder, 2004).

Figure 2.6: Example of a Programmable Logic Controller (PLC)

Linear Actuator

The linear actuators (*vide* Figure 2.7) are used in active one/two axis solar trackers to move the axis in order to track the sun's position. They are able to sustain the height of the structure that holds the solar panel(s) in position while being able to sustain the atmospheric conditions (wind, rain, etc). They offer strength, robustness and reliability reason why they are the most widely used equipment to move the solar panels in the industry. Can either be powered by AC or DC voltages and offer a wide range of supply voltages.

Figure 2.7: Example of a Linear Actuator

Variable Frequency Drive (VFD)

A VFD (*vide* figure 2.8) allows to drive AC motors, in this case an AC linear actuator, by varying the motors input frequency and voltage, greatly simplifying its use.

Figure 2.8: Example of a Variable Frequency Drive

They allow DC to AC and AC to AC configurations enabling a wide range of applications while providing the following advantages: energy saves, low motor starting current, reduction of thermal and mechanical stresses on motors and belts during starts, high power factor and lower KVA (Carrier, 2005, p. 4).

Luminosity Sensor

In solar trackers that use an active tracking mechanism, the luminosity sensors track the maximum point/position of energy output by reading various sensors, which range from LDRs, that are simply light dependent resistors, to digital luminosity sensors that convert the sun's luminosity received into to the corresponding SI unit called Candela.

These sensors can be used in different ways:

- In one axis solar trackers, two luminosity sensors can be placed along the moving axis to determine the differential between those two sensor and act to eliminate the differential. In two axis solar tracker a larger number of sensors is needed, at least four, and the differential is calculated along the two moving axis.

- The shade balance principle can also be applied (Mousazadeh et al., 2009, p. 1807). It consists in using an opaque plaque in order to create a difference in luminosity in the sensor cells.
- Another configuration uses a tilted mounted sensors that allow to ascertain the relative position of the sun based on the difference between the two measuring sensors (Mousazadeh et al., 2009, p. 1807).

They are also used to detect the situation of a cloudy day. In a cloudy day, the point of maximum luminosity may not be the apparent position of the sun due to the existence of refractions of light within the clouds. This fact ensures that a solar tracker that uses a luminosity based controller is always tracking the position that ensures that maximum performance.

Anenometer

To maintain the structural integrity of the solar tracker structure, it is added an anemometer, (*vide* Figure 2.9) which is used to measure the wind speed, so that the structure can react to strong gusts of wind that could damage the structure. A safe position is defined so that the wind resistance of the structure is minimized and when the wind speed reaches the given upper threshold the structure moves to the aforementioned safe position.

Figure 2.9: Example of a Anemometer

Real Time Clock

One of the widest used components is the RTC, that is present in everything from mobile devices, personal computers to autonomous systems. The RTC tracks the current time and provide it via a serial/parallel communication. Usually also provides a backup mechanism that enables to continue to function when the master system is not powered or even during power failures. Some RTC's also provide a few bytes of RAM backuped memory for retaining important information during power failures. The backup power source is usually a coin cell/battery.

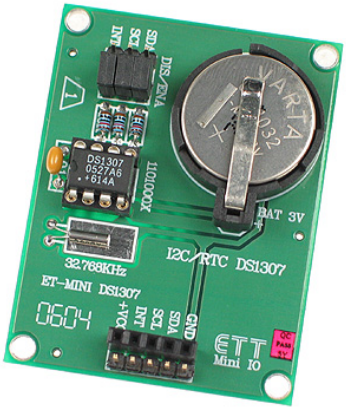


Figure 2.10: Example of a Real Time Clock with serial communication

Chapter 3

Architecture

3.1 Requirements

To define an architecture we first need to specify the requirements that it must meet. As explained in the Chapters 1 and 2, there is a multitude of different types of solar trackers available in the market and to try to create an architecture that would be able to control all these different types would be infeasible in this dissertation.

With the former argument in mind, two types of requirements were defined:

- System Independent Requirements
- System Dependant Requirements

3.1.1 System Independent Requirements

These requirements were specified without looking at any prototype or system in particular and must be regarded as a set of rules that the controller created must meet in order to differentiate itself from the solutions in the market.

Market Solutions

In order to differentiate our solution from the ones existent in the market, we must first do a market survey of all the solutions available to consumers and enterprises and divide them by relevance to our specification.

The top of line solutions from Siemens are considered to be the most reliable and the *SIMETIC S7-1200* is the latest model for solar solutions (Siemens, 2014). It uses the PSA algorithm to calculate the sun position and a set of high precision gears to obtain an accuracy rate of 0.003%. Despite this advantages it comes with a hefty price of, approximately, 500€(RS, 2014). For programming the PLC it requires the *SIMATIC STEP 7* proprietary software which, if not included at purchase, can cost up to another 500€.

From mid tier manufacturers of PLCs we can find a wide range of solutions that offer low prices but do not offer the same reliability and prestige of the Siemens solutions, like Yokogawa's HXS10-SolStation (Yokogawa, 2014) (no pricing available) or the Hitachi EH-WD10DR with its internet capabilities (sending emails responding to various triggers and web interface).

There is also a segment of the market in which the solutions are provided by low quality manufacturers whose solutions are closed without the possibility of programming modifications or adding new functionalities. They are usually very cheap, in the range of the 100€but do not offer any assurance of reliability, such is the case of *Fusionseeker* and *Heliotrack*.

Requirements

As aforementioned, Section 3.1.1, these controllers usually are expensive, programmed using proprietary software, which is also expensive, and they usually have limited features and costly upgrades. The cheaper ones do not offer as much options and the reliability is, at most, questionable.

To address these problems and to increment value to the proposed solution, we must attend to the following questions:

- - The designed controller must be cheaper than the commercial solutions available in the market while being, at least, as reliable as they are;
- - The controller should implement the tracking algorithm with an accuracy of, at maximum, 1° from the real sun position;
- - The controller must work with the existent tracker hardware, therefore being compatible with the PLC solution;
- - It must work using the power source available at the solar tracker without any change in the structure;
- - Interface connection must be simple, reliable, cross-platform and open-source;
- - Addition of new features must not require proprietary software and be simple and easy to configure.

3.1.2 System Dependant Requirements

As mentioned earlier, there is a vast number of different configurations of solar trackers in the market (*vide* Section 2.2). Given this fact, the necessity of defining a configuration that our architecture will be designed to control is paramount. Without this specification, there are too many different kinds of inputs and signals that would make the architecture overly complex to create.

To define the requirements we used a prototype, named *SPH 4.0*, created by a local company, *Sunshpere*, which allowed us to have a reference model, whose capabilities and limitations are known to us, in order to create the architecture and, eventually, be able to test the controller created.

The characteristics of the prototype will be fully detailed in the following sections.

Prototype Characteristics

Axis	Azimuth	Zenith
Range	-135 to 135 degrees	0 to 90 degrees
Actuator type	Rotation Gear	Linear Actuator
Rated Power	150 W	90 W
Voltage Supply	220 V	
Maximum Wind Speed	80 Km/h	
Tracking Mode	Chronological	
LDR	Yes - One	

Table 3.1: Solar Tracker Characteristics

From Table 3.1, we know that the prototype has a 270 ° range in the azimuth axis and a 90 ° range in the zenith range. In order to be efficient, the controller created must be able to fully control the prototype in the range of the structure.

Must also be able to provide the necessary AC power required by the VFD to move the axis actuator, 150W and 90W at the azimuth and zenith axis repectively. The controller must also be able to measure the wind speed in order to protect the structure in cases that where the limit speed was surpassed, moving the prototype into a safe position.

Given the existence of only one light/luminosity sensor, the controller must use a chronological algorithm to predict the position of the sun and use the existent LDR to detect a cloudy day and maximize the efficiency in that situation.

PLC Controller

The prototype uses a PLC to control the solar tracker, using a set of inputs and outputs. We will use the parameters of these I/O to specify the I/O's required by our control board. By using these I/O's we can think of the structure as a black box. The black box concept identifies that the overall structure can be simplified via its inputs and outputs and their functionality regardless of how they are connected within the structure. Tables 3.2 and 3.3 describe the inputs and outputs of the current PLC implementation.

Outputs

Outputs			
Designation	Function	Type	Voltage Range (V)
O0	VFD Enable 1	Analogue	220 AC
O1	VFD Enable 2	Analogue	220 AC
O2	VFD Direction 1	Digital	0 - 24 DC
O3	VFD Direction 2	Digital	0 - 24 DC
O4	VFD Speed	Digital	0 - 24 DC

Table 3.2: PLC Outputs

The five output ports from the Table 3.2 connect directly to the VFD to enable the control of the axis actuators. The DC outputs (O2 through O4) must be left in a pnp configuration due to the prototype specification.

Outputs *O0* and *O1* enable the VFD for each of the axis. The VFD is also powered from these output ports, reason why they must be able to provide up to 150W (minimum) each (*vide* Table 3.1).

O2 and *O3* control the direction of each of the axis while *O4* controls the speed of the VFD. The speed, in a VFD is a multiple of the frequency of the power grid (50Hz), in this case can be 50Hz or 100Hz according with the state, 0 or 24V, of that output.

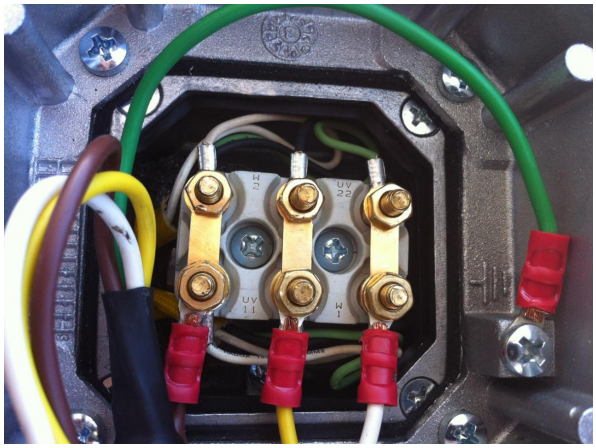


Figure 3.1: VFD terminals of the prototype

Inputs

Inputs			
Designation	Function	Type	Voltage Range (V)
D0	Anemometer - High Frequency Counter	Digital	0 - 24
D1	Optical Encoder - Azimuth	Digital	0 - 24
D2	End Course (Azimuth Calibration)	Digital	0 - 24
D3	End Course (Azimuth Emergency)	Digital	0 - 24
D4	End Course (Zenith Top)	Digital	0 - 24
D5	End Course (Zenith Bottom)	Digital	0 - 24
D6	VFD Alarm	Digital	0 - 24
D7	Emergency Button	Digital	0 - 24
A0	Zenith Inclination	Analogue	0.5 - 4.5
A1	LDR Sensor	Analogue	0 - 5

Table 3.3: PLC Inputs

The eight digital and two analog input port enable the controller to read the various states of the different components in the solar tracker.

The input *D0* is connected to anemometer which via a high frequency counter is capable of measuring the wind speed in the vicinity of the solar tracker.

The *D1* input is wired to a optical encoder, Figure 3.2, for measuring the relative position of the azimuth axis to the calibration point, Figure 3.3, given by the input *D2*. By converting a single step of the encoder to the difference in the azimuth angle is possible to measure the relative angle of the solar panel. The *D3* input is connected to an end course switch in the azimuthal axis to prevent movements of the axis over the allowed range, Figure 3.4.

Figure 3.2: Azimuth Encoder

Figure 3.3: Azimuth Calibration End Course

The inputs *D4* and *D5* are connected to the zenith axis end courses to prevent movement outside of the 0 to 90 degrees range, Figure 3.5.

Figure 3.4: Azimuth Emergency End Course

Figure 3.5: Zenith End Courses

The VFD alarm input is wired directly to the VFD and its activated when some malfunction is detected, while the Emergency Button is connected to the box enclosing the controller and it is accessible to the owner. It is meant to stop the solar tracker altogether, Figure 3.6.

Figure 3.6: Emergency Button

3.2 Proposed Architecture

To address the requirements identified in Section 3.1, the following architecture is proposed:

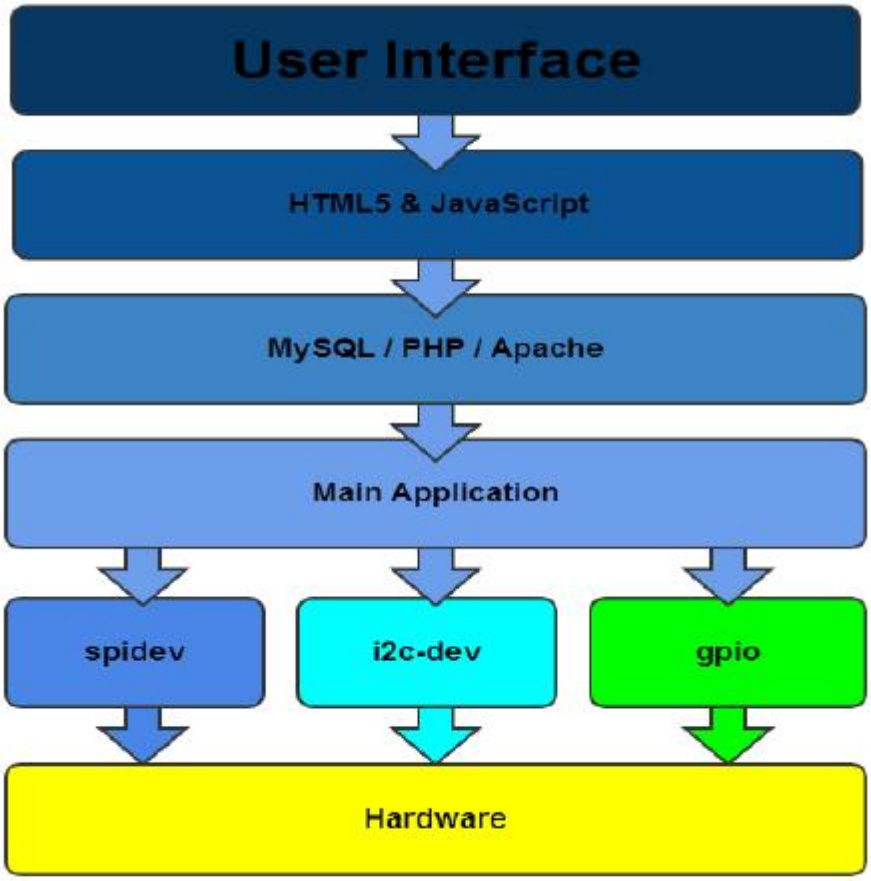


Figure 3.7: Proposed Solution Architecture

The aforementioned architecture, figure 3.7, will be implemented using a Raspberry Pi single-board computer running a port of Debian to ARM devices. It will be responsible to implement all the control algorithms and the user-interface via Ethernet port. By implementing the user-interface via a web browser we are eliminating the need for custom user interfaces for each of the mainstream OS'es (Windows, OSX and Linux) and their numerous versions.

Because the Raspberry Pi was built with input/output ports we can, within limits, interface it with various types of devices. However, the GPIO's are not protected and should only be interfaced with devices having a voltage supply of 3.3 volts. For this reason an external interface board must be used to safely interface the Raspberry Pi and the solar tracker prototype (vide section 3.1.2).

In order to control the prototype, a main application will be created, and this application is responsible to implement the tracking algorithms, log the system parameters to the database, in MySQL, and control the various devices in the interface board (mainly via I^2C and SPI

communication) and the Raspberry-Pi GPIO's. This main application will be created in C++ language.

A middleware will be created to interface the main application and the user interface, recurring to PHP for database query's from the HTML5 user interface.

The user-interface will be coded in HTML5 and Javascript and will be deployed in an Apache HTTP local server. This enables the user to access the user interface by just typing the IP address of the Raspberry Pi. The user-interface should also be able to implement the following features:

- - Maintain and display detailed logs with the most important info about the solar tracker;
- - Manually change the solar tracker position;
- - Implement user authentication to prevent unauthorized access;
- - Display general information about the controller (system up-time, temperature, etc);
- - Work with the current browsers available in the market.

The hardware and software implementation will be deeply explained in the chapters 4 and 5, respectively.

Chapter 4

Hardware

As referred in the section 3.1, the proposed controller must work with the testing system power supply and input/output ports. For that reason an interface board was designed and implemented. This chapter will fully explain the design and functionality of the implemented interface board.

The Raspberry Pi, figure 4.1, is a single-board ARM computer clocked at 700 MHz running a ported distribution based on the Debian OS (Linux based). It features 26 GPIO's for interfacing with external devices. Some of the GPIO will be used to interface the controller.

a) b)

Figure 4.1: a) Raspberry Pi model B - b) Raspberry Pi GPIO description

In order to simplify the explanation of the interface board, the presentation was divided different into fundamental parts, which are explained separately:

- - DC to DC converter - used to provide the necessary voltage supply to power the Raspberry Pi and the other devices used;
- - Analogue to SPI interface - used to convert the analogue voltage of the Zenith Inclination and the LDR sensor to a serial format;

- - I/O Expander by I^2C with optical coupling - To interface the Raspberry Pi with the 24 volts rated inputs/outputs;
- - Optical Coupling to TRIAC switch - Used to switch on/off the 220 volts AC rated VFD enables;
- - Voltage Converters from 5 volts to 3.3 volts - Used to protect the Raspberry Pi from 5 volts rated signals (SPI and I^2C);
- - RTC with battery backup - needed to hold time and important info in an event of power failure;
- - Temperature Sensor with I^2C communication - needed to register the controller temperature and warn in an event of a over-temperature situation.

Which resulted in the following block diagram:

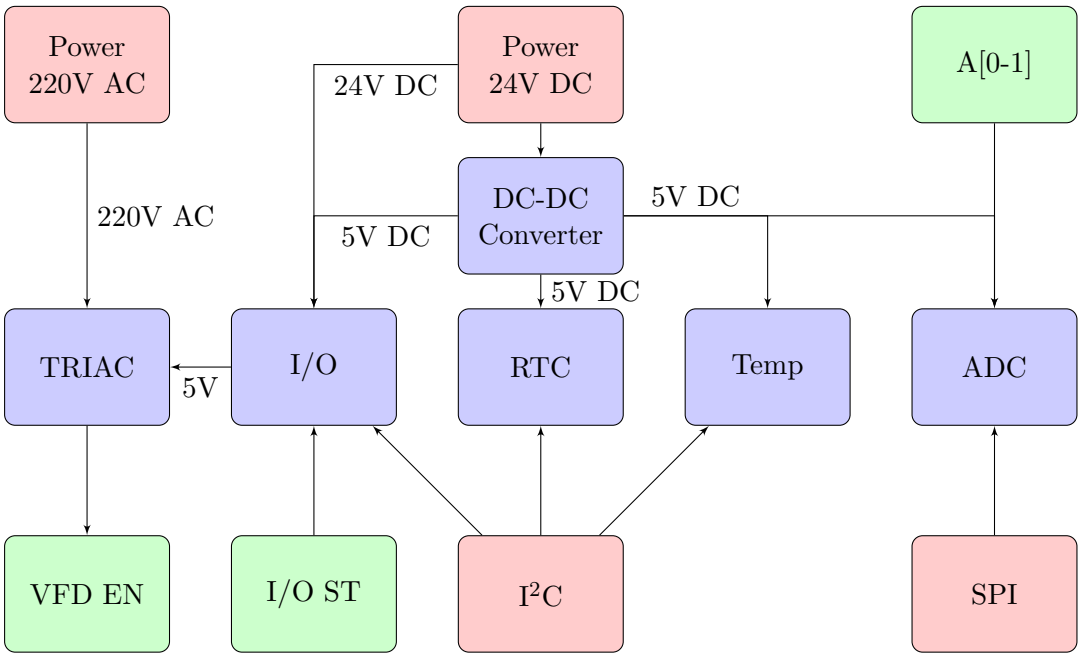


Figure 4.2: Interface Control Block Diagram

In the Figure 4.2, blue block represent the different components of the interface board, the green ones represent the connections with the solar tracker, namely the VFD controls and the inputs from the sensors. Red blocks represent connections with the Raspberry Pi or with the power supply. All these components will be fully explained in the next sections.

4.1 DC to DC converter

In order to implement the necessary step-down converter from 24 volts to 5 volts needed by the Raspberry Pi an MC34063 DC to DC controller from Texas Instruments was used. This

controller simplifies the design and operation of a converter by implementing the necessary control feedback and current limitation in-package.

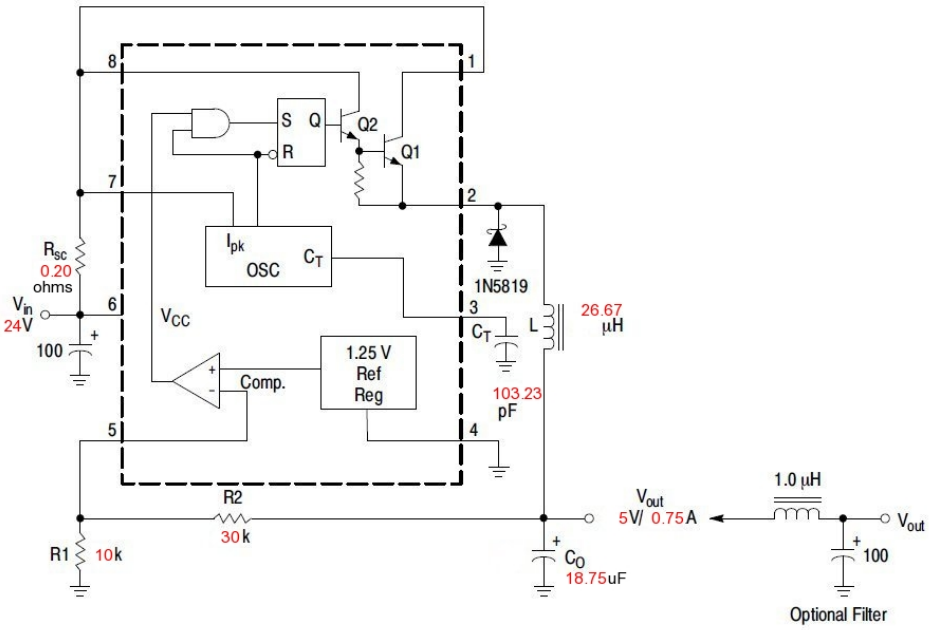


Figure 4.3: Implemented Step-Down Calculations

These results were reached following the equations in (MC34063A, 2010) which are shown in the table 4.1.

Calculation	Equation	Result
$\frac{t_{on}}{t_{off}}$	$\frac{V_{out}+V_F}{V_{in(min)}-V_{sat}-V_{out}}$	$\frac{5+0.6}{23-1.3-5} \approx 0.336$
$(t_{on}+t_{off})$	$\frac{1}{f}$	$\frac{1}{100khz} = 10*10^{-6}s$
t_{off}	$\frac{t_{on}+t_{off}}{\frac{t_{on}}{t_{off}}+1}$	$\frac{10*10^{-6}}{1.336} \approx 7.49*10^{-6}s$
t_{on}	$(t_{on}+t_{off})-t_{off}$	$(10-7.49)*10^{-6} = 2.51*10^{-6}s$
C_T	$4*10^{-5}*t_{on}$	$4*10^{-5}*2.51*10^{-6} \approx 100*10^{-12}F$
I_{pk}	$2*I_{(out)max}$	$2*0.75 = 1.5A$
R_{SC}	$\frac{0.3}{I_{pk}}$	$\frac{0.3}{1.5} = 0.2\Omega$
L_{min}	$(\frac{(V_{in(min)}-V_{SAT}-V_{out})}{I_{pk}})*t_{on}$	$(\frac{(23-1.3-5)}{1.5})*2.51*10^{-6} = 27.95*10^{-6}H$
C_O	$\frac{I_{pk}*(t_{on}+t_{off})}{8*V_{ripple(pp)}}$	$\frac{1.5*10*10^{-6}}{8*0.1} = 18.75*10^{-6}F$

Table 4.1: DC to DC converter equations

To further reduce the strain in the device, a bigger inductance L was used, $L = 100\mu H$, resulting in a lower current ripple in the inductance, thus requiring less peak current from the

device, maintaining it in the workable region. To minimize the voltage ripple in the output a capacitor of $470\mu F$ instead of a $18.75\mu F$ capacitor.

The L used was rated up to 1.1 Amps, which means that the DC to DC converter will maintain its stability up to this output current. This means that the interface board plus the Raspberry Pi must only consume up to this limit.

By measuring the current drawn only by the Raspberry Pi when running an Apache Web Server we reached a consumption of, approximately, 2.5W which means a median consumption of 500mA. By adding the consumption of each of the components in the interface board with MCP23017 rated at 1mA@5V, DS1307 also rated at 1mA@5V, LM75A rated 5mA maximum, logic converters up to 12mA each, MCP3002 with 0.5mA and each optic coupler with 2mA summing up to a total of approximately 570mA which is lower than the limit previously calculated. All these rating where retrieve from the fabricators datasheet.

4.2 Analogue to SPI interface

Since the Raspberry Pi does not have an ADC and this is a feature needed to control the solar tracker, it was added to the interface board a MCP3002 Two-Channel, 10 bit ADC with SPI communication from Microchip, Figure 4.4.

This enables the conversion of both the voltage of the Zenith Inclination and the LDR sensor using a single device.

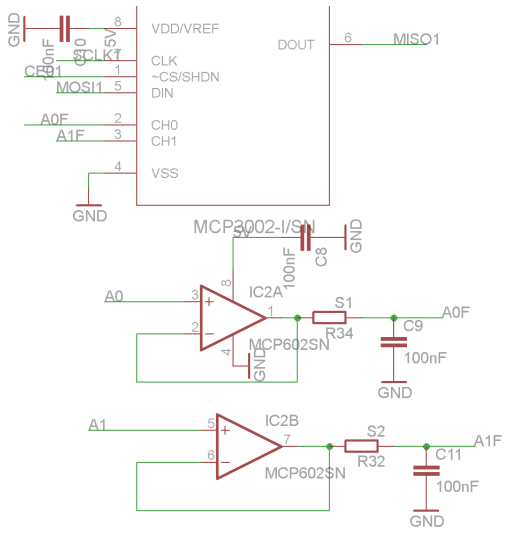


Figure 4.4: Dual Channel SPI ADC with low pass filter

To mitigate the interference received from other devices, a low pass filter was applied using an opamp, a series resistor ($R = 30K\Omega$) and a parallel capacitor ($C = 100nF$) to create a cut frequency of, approximately, $50Hz$, with $f_c = \frac{1}{2*\pi*R*C}$. The addition of this low pass filter stabilizes the voltage read by the ADC, especially from the interference of the power source of the sensor to read, *i.e.* the $50Hz$ frequency of the power grid. Since the expected variation of the analogue value to be read between samples is small, the addition of the filter does not delay the signal significantly.

4.3 I/O Expander by I^{2}C with optical coupling

This section is divided into two fundamental parts:

- - Optical Coupling for voltage protection;
- - I/O Expander via I^{2}C communication.

Since the signal operating voltage for interfacing the solar tracker actuator is 24 volts DC and the controller operates at 5 volts it is necessary to add level-converter hardware. A logic "1" level corresponds to, approximately 24V and 0 volts to the '0' level. As the Raspberry Pi is unable to handle such type of voltage and the usable number of pins is not enough for all inputs and outputs required, an external I/O expander was used. The choice was the MCP23017 16 port I/O Expander with I^{2}C communication from Microchip.

I/O Expander

The MCP23017 can work within a voltage supply range 1.8V to 5.5V and is capable of signaling configurable interrupts to a microprocessor via INTA and INTB pins. These interrupts are cleared via a read of the register by I^{2}C. The MCP23017 has 3 address pins enabling up to 8 chips in the same I^{2}C bus.

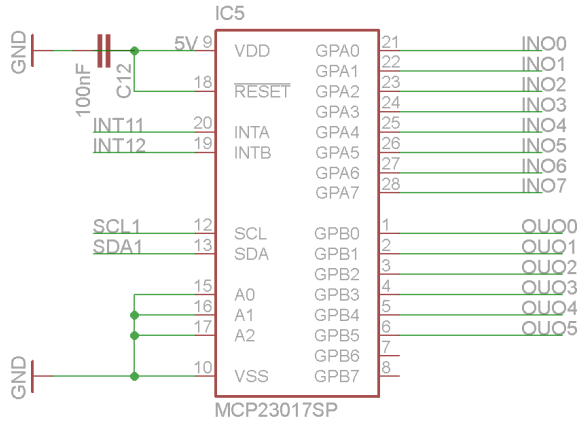


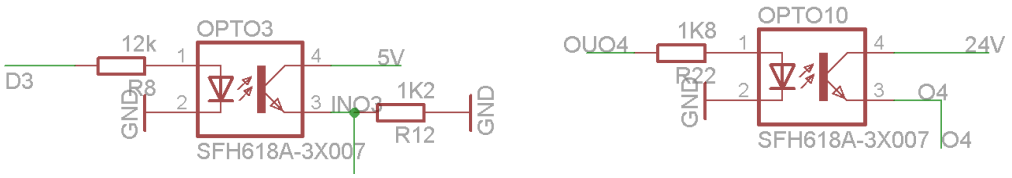
Figure 4.5: 16 port I/O Expander Connections

The connections made between the inputs and outputs of the interface board and the solar tracker are shown in the figure 4.5.

Otpical Coupling

In order to protect the interface board from the 24 volts signal a set of optical couplers were used, namely the SFH618 from *Vishay*. From (Vishay, 2014) datasheet we know that for I_F of 5mA we have a typical V_F of 1.1V.

In Figure 4.6 a), when the input is 24 volts, a current of $\frac{24-V_{(F)LED}}{12000\Omega} \approx 2mA$ turning on the LED inside of the optocoupler. This closes the photo-transistor, connecting $5 - V_{SAT}$, with $V_{SAT} \approx 0.4V$, to the input of the i/o expander. When the input is at 0 volts, the LED is turned off so the photo-transistor is opened and the input is tied to ground.



a) Input Optical Coupler b) Output Optical Coupler

Figure 4.6: Input and Output optical connection

In Figure 4.6 b), the same functionality occurs. When the output of the i/o expander is high, the current of $\frac{5-V_{(F)LED}}{1800\Omega} \approx 2mA$ turns on the LED and closes the photo-transistor connecting the output to 24 volts. The output is left in a pnp configuration due to the solar tracker specifications.

Due to the usage of the optical coupling in the interface board, there is a theoretical limit of the frequency that the input/output signal can reach. This frequency can be higher or lower depending on the I_F current, I_C current and the capacitance of the lines. By increasing I_F we also increase the overall consumption of the interface board but the maximum frequency increases. If we lower I_F we decrease the overall consumption but we also decrease the maximum frequency of the input/output signal. A compromise was made to minimize the power consumption while allowing a maximum switching frequency sufficient to handle the signals from the solar tracker, namely from the encoder and anemometer. By using the I_F previous calculated we reached a tested maximum switching frequency of $2KHz$ which is sufficient to handle the low frequency signal from the optical encoder.

4.4 Optical Coupling to TRIAC switch

The VFD enable outputs of the interface board are 220 V AC capable of powering the rotation gear and the linear actuator, 150 and 90 Watts, respectively. To ensure that such high power does not negatively impact the interface board it was implemented a optical coupling tied to a TRIAC, shown in figure 4.7. The selected circuit is widely used when interfacing digital systems with analogue switches and will be only briefly explained. The implementation will follow the inductive load approach shown in (Instruments, 1998).

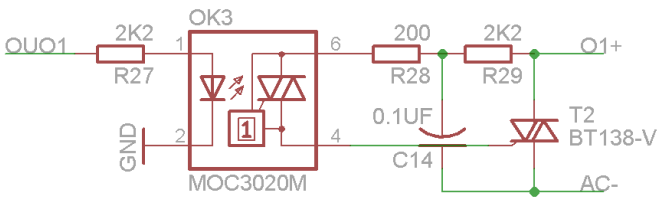


Figure 4.7: MOC3020 with a BT138 TRIAC

Since this is widely used circuit, the only detail need explanation is the importance of the capacitor, in the figure 4.7 named $C14$. Since this circuit does not have a zero-cross detection, and thus being switched on or off at any given moment, this capacitor should be

able to sustain rated voltages of 400 volts to ensure that the spike created by the turn on/off action of the induction motors, which rotation gears and linear actuator are example off, does not damage the circuit, namely the $2.2K\Omega$ resistor.

4.5 Voltage Converters

Since the Raspberry Pi is rated at 3V3 volts and all other devices are rated at 5 volts, to ensure proper communication between the devices two voltage converters were used:

- - TXS0102 - Voltage Translator from Texas Instruments appropriate for I^2C enabled buses. Eliminates the need for external resistor pull-ups since it has internal pull-up resistors.
- - TXB0108 - Voltage translator from Texas Instruments used for translating SPI communication and interrupt signals from 5 volts to 3.3 volts and vice-versa.

These two converters implement all the need hardware from translating the signals within the ranges needed and without any needed for supplying the direction of the information transfer.

4.6 RTC with battery backup

The chronological tracking algorithm accuracy depends on the exactness of the controller time. For reliability and in an event of power failure the system must be able to resume the functionality and restore the exact time. For this purpose, it was added to the interface board a real time clock with battery backup RAM, figure 4.8. The DS1307 accurately maintains the time and also provides 56 bytes of battery backuped RAM so that important data can be stored and restored in case of power failures.

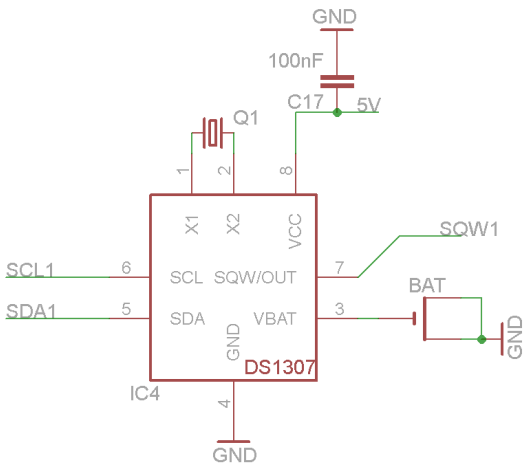


Figure 4.8: DS1307 RTC with battery backup

This RTC communicates via I^2C and it is powered at 5 volts. The battery used it is a CR1220 coin cell rated at 3 volts.

4.7 Temperature Sensor via I^{2}C

Since this controller will be exposed in the outside it is important to track the controller temperature due to various circumstances:

- - Ensure that the DC to DC converter works within the specified temperature range avoiding over dissipating power leading to the destruction of the MC34063 chip;
- - Possibility of turning off features that generate heat for cooling the controller.

A LM75 chip was selected because of its ability to communicate via I^{2}C, resolution of 0.5 °C, accuracy of ±2 °C in the 25 °C to 100 °C temperature range. The LM75 has 3 address pins enabling up to 8 chips within the same I^{2}C bus.

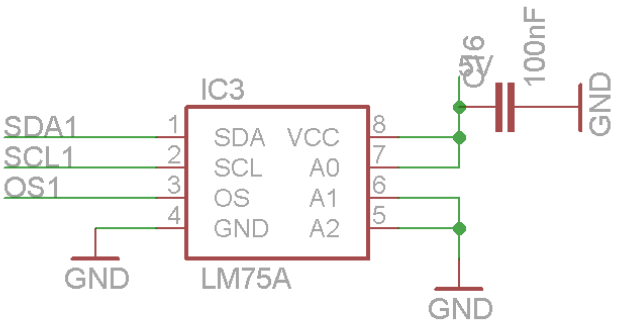


Figure 4.9: LM75A Temperature Sensor

Figure 4.9 identifies the connections made within the interface board to enable the reading of the LM75A temperature sensor.

4.8 Addendum to the initial board

The board created relied on the availability of interrupts in the Debian port (Raspbian) of the Raspberry Pi. Since currently there is no support for hardware interrupts on the Raspberry Pi, a PIC32MX220F032B was added to handle the anemometer and encoder interrupts (frequency counters) the communication of these values to the Raspberry Pi is made via the I^{2}C bus, *vide* Figure 4.10.

Another change made was due to the fact that the TXS0102 I^{2}C logic converter used not perform as expected. So the IC was removed and the two N-FET transistors follwing NXP AN10441 application note were used.

While testing the hardware an interference was detected between the optical couples and the TXB0108 logic converter, which impeded the signal to fall to a zero logic state. While the reason for this to happen was not found, the problem was solved by using an OPAMP in buffer configuration between the optical coupler and the logic converter.

Figure 4.10: Addendum to the initial board

Chapter 5

Software

The following chapter will explain in detail the software made for interfacing with the board created and controlling the solar tracker. First is given an justification for choosing the MySQL database to store the information created. Then a brief overview of the functionality of the hardware device driver will be given along with the state machine implementation and diagram. The main application flow of execution along with its various options will be detailed and the user interface will be shown.

The hardware device-drivers and the state machine was coded in c++ and compiled using the CMake cross-compiling platform for ease of use.

5.1 MySQL

During the execution of the main program it is important to log the necessary information to enable the detection of malfunctions in its execution. Given that a solar tracker must function during extended periods of time, the size of information created would impact the performance of the program if it was stored directly on the program structures.

Since we must also create an user interface it is crucial that the information created is reliable and not subject to tampering.

The MySQL database was chosen due to its native support in Linux and its easiness of use, while having the necessary connectors for C++ language and simple yet powerful operations via PHP language enabling its usage as source of information to the user interface. So, the MySQL database will act as a link between the information produced by the main algorithm and the user interface requests.

5.1.1 MySQL tables and C API

In order to hold the information a series of MySQL tables were created, see figure 5.1.

Each of the tables enable the implementation of the features specified in the section 3.1.

BootLog

The **BootLog** table information is filled at the start of the solar tracker class, namely by a call to the configureSolarTracker function. The information will be used to track the system up-time, initial temperature, position and mode of operation to provide it to the user via the web interface.

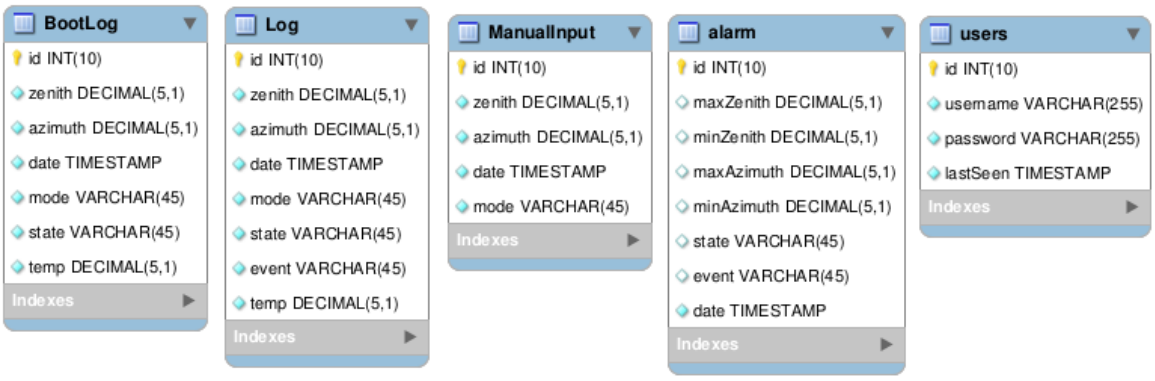


Figure 5.1: MySQL tables created for the Web Interface

Log

The **Log** table information is filled at each change of the state machine or mode and it is the primary source of information to identify if the program is behaving as expected. It enables to implement the log viewer via web interface and to identify if a power outage has occurred by comparing the last entry in the log and the current information.

ManualInput

The **ManualInput** table enables the insertion of a manual set point (azimuth & zenith) in the web interface and the respective synchronization with the main program. The main program checks regularly for any change in this table and if there is a new set point, the various rows are read and then deleted thus simplifying the discovery of new information.

Alarm

The **alarm** table enables the insertion of an alarm or the execution of a set of instructions triggered by a value outside the permitted range. Although this kind of triggered events was not used in the main program, the necessary functions to communicate with the table were made opening the possibility of an later update.

Users

The table **Users** contains the information regarding the authorized users that can access the advanced feature of the web interface, such as manual positioning, log delete and alarm setting. The username cam be either the name of the person or the email in the company while the password can be anything up to 255 characters. The lastSeen info gives the information about the last time this user authenticated in the web interface.

5.2 Hardware Device-Drivers

For each of the devices present in the interfacing board was created an c++ class which exposes a set of public functions to the main controller program while maintaining private

information.

5.2.1 MCP3002 - ADC

The MCP3002 device driver is responsible to convert the input analogue value from the LDR and the *Zenith* sensor. In order to accomplish this, the driver has the following execution diagram:

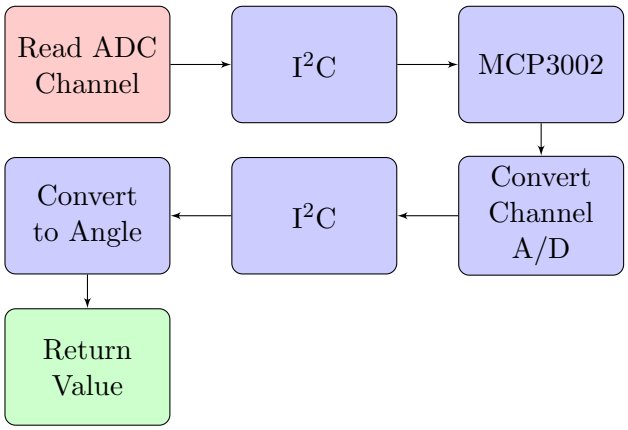


Figure 5.2: MCP3002 Device Driver Execution Diagram

Since the MCP3002 has two analogue to digital conversion channels, the device driver is capable of converting all the existing analogue signals and returning them to the main application.

As the MCP3002 is powered at 5V, the range of conversion is, approximately 5V and the resolution of conversion, with this being a 10 bit ADC, is $\frac{V_+ - V_-}{1024} = \frac{5-0}{1024} \approx 0.005V$. If we translate the same principle to the angle range we get a angle resolution on the *Zenith* axis of $\frac{90}{1024} \approx 0.09°$. The algorithm to convert from the digital value retrieved by the ADC to the *Zenith* angle is $Z_{angle} = D_{value} * 0.09$ from $Z_{angle} = V_{CH} * \frac{2^{10}}{V_+ - V_-} * \frac{90}{2^{10}}$ resulting in $Z_{angle} = V_{CH} * \frac{90}{5}$, with V_{CH} being the analogue value at the input channel of the ADC and $V_{CH} * \frac{2^{10}}{V_+ - V_-}$ the conversion to the [0-1024] digital range. To further eliminate any noise from the value read, an mean filter of 10 samples was applied. Given the high rate of sampling of the ADC, approximately $100KHz$, no significantly delay is added to the operation.

5.2.2 MCP23017 - I/O Expander

The MCP23017 device driver must be able to effectively control all the required inputs and outputs from and to the solar tracker. To accomplish this the device driver must be able to change the state of a given pin while maintaining the others in the same state. This is called a ***read-modify-write*** execution due to the fact that it is only allowed to write the state of all pins in a port (8 pins in a port) and it will be explained in Figure 5.3. It must also be able to read the state in a given port, if it was configured as input.

The interrupt function of this I/O expander will not be used due to the lack of proper hardware support of interrupts in the Raspberry Pi Linux kernel.

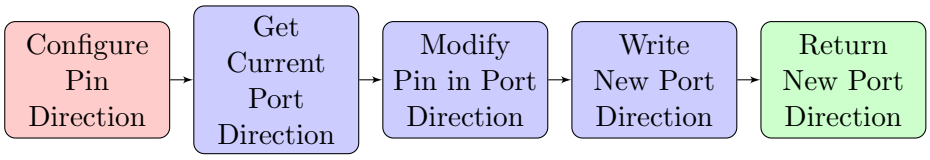


Figure 5.3: MCP23017 Device Driver Execution Diagram

The process is similar when changing a given pin state configured with output direction. Any change in a given pin state must not interfere with the states configured in the adjacent pins. When reading a given pin state, when configured as input first we read the entire port status then the single pin is matched against the port state via an AND logical operation resulting in the state of the pin. When requesting an operation with the wrong pin direction configured the result will be set to null and an error will be returned to the main program.

5.2.3 LM75 - Temperature Sensor

The LM75 device driver allows to monitor the temperature of the interface board to prevent over-temperature situations, using the execution diagram in Figure 5.4.

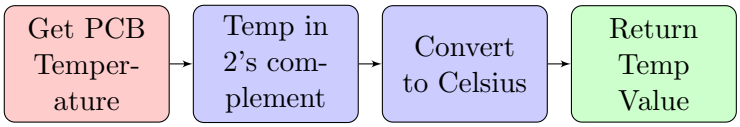


Figure 5.4: LM75 Device Driver Execution Diagram

The temperature is stored in 2's complement, which is a way to code singed numbers into a number of bits by subtracting the desired negative value from the maximum range of the normal unsigned number, *i.e.*, -7 in 8-bit two's complement notation is equal to $2^8 - 7 = 121_{10} = 1111001_2$ while the representation of the number 7 is equal to the standard form, 0000111_2.

The LM75 stores the temperature in two bytes, one for storing the integral part in 2's complement and the other for the fractional part, where only the MSB is relevant and should be multiplied by $0.5°C$ to calculate the final temperature, resulting in a resolution of $0.5°C$.

The LM75 also allows others configurations namely an auto overtemperature interrupt output but it was not used because of the lack of interrupt support.

5.2.4 DS1307 - RTC and NVRAM

The DS1307 device driver is crucial to the correct execution of the solar tracker. Given the usage of the chronological algorithm to track the sun position, any deviation from the correct time results in discrepancies in the final calculated position resulting in a loss of performance.

It is also importance in the occurrence of a sudden power failure because the information in the database might become corrupted. In this case, the RTC holds the last important information enabling the restoration of some variables, namely the zenith and azimuthal

position, eliminating the need to recalibrate both axis. Figure 5.5 represents the execution of the DS1307 device driver when retrieving the time from the device.

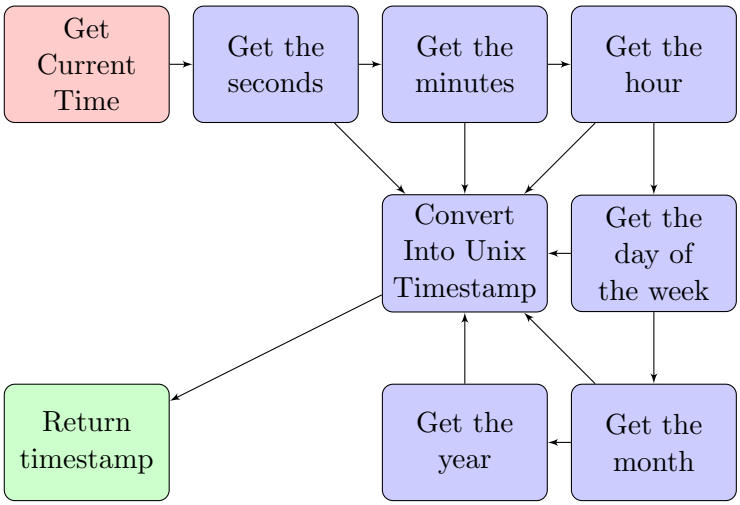


Figure 5.5: DS1307 Device Driver Execution Diagram

Figure 5.5 shows the how the DS1307 device driver retrieves the time in the device. Since the device stores the data time in several bytes in different address, which one representing a particular subset of the time representation, the application must read all these bytes from the device and converter them into a UNIX timestamp.

To enable storing of variables larger than an byte size, a conversion and store method was implemented responsible to store the variable in sequential addresses of the NVRAM.

5.2.5 External Interrupt Handler

Since the support for hardware interrupts is limited in the current Linux kernel, a device driver was implemented to communicate with the PIC32MX220F032 to retrieve both the ***encoder*** and ***anemometer*** counters.

The PIC32 program is counting every interrupt for each of the connected signals and stores them in the PIC32 internal memory and can be read by the controller via I^2C, in the defined address.

The Figure 5.6 represents the execution diagram of this device driver.

Since the encoder provided only outputs one signal line, it is not possible to ascertain the direction of the movement. To mitigate this, the value read from the encoder counter is compared to the current configured movement of the solar tracker. Based on this apparent movement is possible to calculate the relative encoder position of the solar tracker in the azimuthal range, which will be later converted to an angle relative position.

5.3 Board Definition Package

The board definition package is the high-level API to be included in the main control program. It is responsible for the following operations:

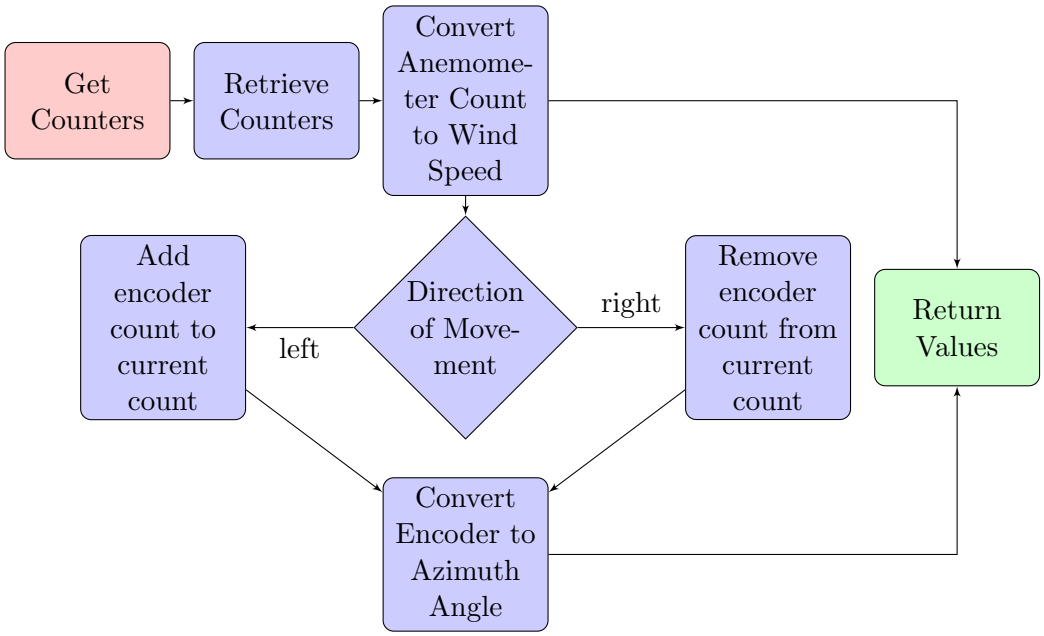


Figure 5.6: External Interrupt Device Driver Execution Diagram

- Update the internal information at a rate of 100 Hz, which, given the expected slow variation of the parameters to read is sufficient to maintain a reliable information of the solar tracker state;
  - Read the counters from the External Interrupt Device;
  - Read the Zenith sensor value;
  - Checks if any end course signal was activated;
  - Check if the emergency button was pressed;
- Initialize all the previous device drivers;
- Hold the current system information;
- Implements the logging functions to the MySQL and RTC NVRAM;

It was created to implement some functions whose operation did not fall in the other device driver characterization.

5.4 State Machine Implementation

The state machine implements the necessary steps to ensure the correct functionality of the solar tracker.

The state machine implements 5 main states:

- Main States
  - Idle

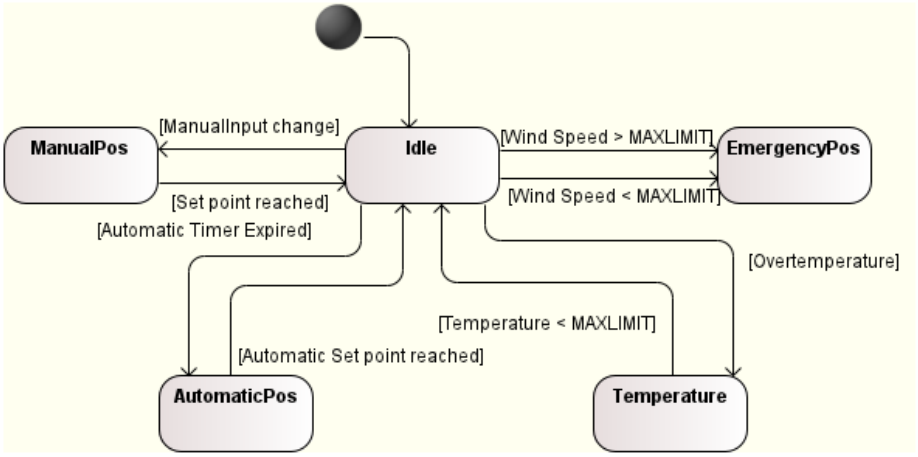


Figure 5.7: Diagram of the implemented state machine

- ManualPos
- AutomaticPos
- Temperature
- EmergencyPos

The **Idle** state implements all the necessary verifications to ascertain the necessity of a state change:

- Temperature of the system - change to **temperature** state;
- Insertion of a manual set point via Web Interface or Command Line Interface - **ManualPos** state;
- Automatic timer expired and a new set point was created - **AutomaticPos** state;
- Wind speed is above the permitted threshold - **EmergencyPos** state;
- Emergency Button was pressed - **EmergencyPos** state;
- VFD alarm went off - **EmergencyPos** state.

The **Temperature** state turns off all outputs in an attempt to reduce the consumption of energy in order to reduce the temperature of the board. This state ensures that components in the interface board operate within their normal operating condition (exceptionally important to the correct functionality of the DC to DC converter).

The **ManualPos** and the **AutomaticPos** states ensure that the manual/automatic position (azimuth/zenith) inserted/calculated is reached. The two states only differ in the signal that activates them while the underlying functionality is the same.

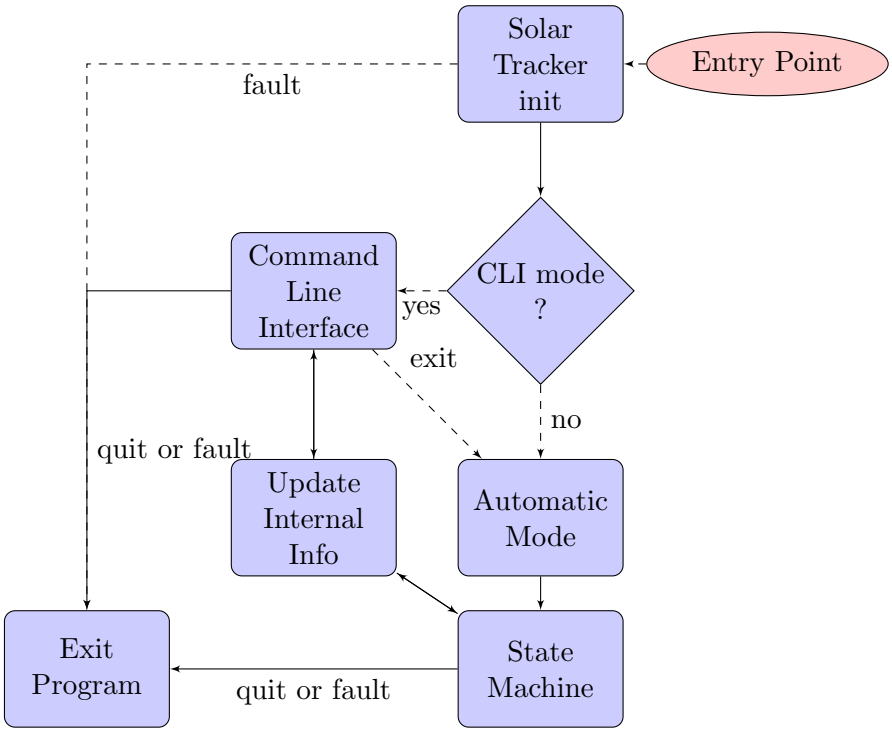
The **EmergencyPos** state is used for three main purposes:

- Positioning the solar tracker in a way that ensures the minimum wind resistance thus limiting the damage made by strong gusts of wind. This position is normally with a zero zenith angle, *i.e.* with the plane of the solar panels parallel to the earth plane.

- Escape state when the user pressed the emergency button (staying in the same state while the button is pressed)
- Escape state when the VFD alarm is triggered due to malfunction. This ensures that the solar tracker functionality is stopped while the alarm is turned on.

When the conditions that caused the state change are cleared, namely the given position was reached or temperature/wind speed is within the correct range, the state machine falls back to the initial state, the **Idle** state, repeating this cycle *ad eternum*.

5.5 Main Program Flowchart



The main program execution flowchart is divided into 3 main stages:

- SolarTracker class initialization
- Default interaction mode choice
- Handle the errors

In the class initialization, all hardware device-drivers are initiated and tested for correct functionality. If an error occurs within this process, the error code is returned to the main program and the execution is halted.

If the prior process is concluded satisfactorily, the program decides whether to enter in the command line interface or to proceed to the automatic mode where the interface with the program is made via web browser. The user provides this information via the arguments when the main program is called. The following arguments are available:

- c - enter command line interface
- a - enter automatic mode
- h - show argument help and exit program

If the chosen mode of operation is the command line interface the following options are available to the user:

- A : MOVE SOLARTRACKER TO THE LEFT
- D : MOVE SOLARTRACKER TO THE RIGHT
- W : MOVE SOLARTRACKER UP
- X : MOVE SOLARTRACKER DOWN
- S : SHOW CURRENT STATUS
- M : MOVE TO DESIGNATED COORDINATES
- C : CONFIGURE SAFETY POSITION
- L : CONFIGURE GEOGRAPHICAL LOCATION
- E : EXIT COMMAND LINE INTERFACE AND EXIT PROGRAM
- P : EXIT COMMAND LINE INTERFACE AND PROCEED TO AUTOMATIC MODE
- H : SHOW COMMAND LINE INTERFACE HELP
- M,C and L require two additional commands
  - M and C : ZENITH AZIMUTH with ZENITH ϵ [0.0;90.0] and AZIMUTH ϵ [-135.0;135.0];
  - L : LONGITUDE LATITUDE with LONGITUDE ϵ [0.0;360.0] and LATITUDE ϵ [-90.0;90.0].

Otherwise if the automatic mode of operation is chosen, the program will enter in the state machine *ad eturnum*, as explained in the section 5.4.

If the user aborts the execution via Ctrl-C and Ctrl-Z commands the program first saves the current state information of the solar tracker to both the MySQL tables, namely to the ***Log*** table and in the RTC RAM and then proceeds to safely terminate the program execution.

In order to keep the information about the state of the solar tracker updated the setitimer and signal functions were used to create a timer of 10 milliseconds of interval, thus creating the $100Hz$ update rate referenced before, recurring to the SIGALARM signal exposed by the linux distribution. This timer is responsible for calling the updateInfo() function responsible for updating the internal information.

5.6 Web Interface

The web interface comprises into 5 different pages which will be explained in the next sections. It uses the *Bootstrap* framework to adapt itself to the resolution of the browser.

5.6.1 Home

The home page is where the user can log in, see the more important info about the solar tracker and navigate to the other pages *vide* Figure 5.8.

This page provides information about the turn on date, the current system time and the time since it was started. It also displays the current *Zenith* and *Azimuth* angles and the current system temperature and a link that take the user to the configuration page for further detail.

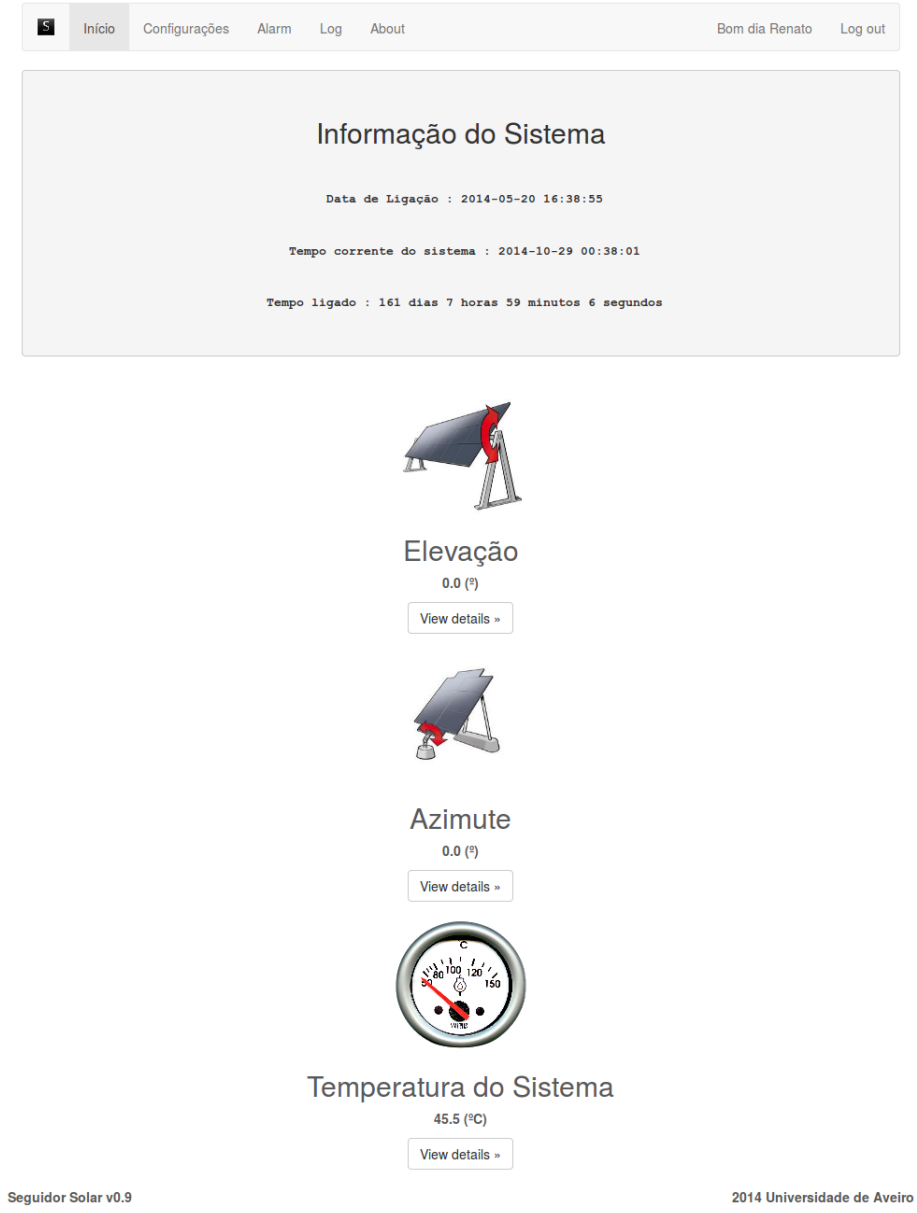


Figure 5.8: Initial Web Interface Page

LogIn

The log in form is reached through a hidden page by selecting the appropriate button on the home page. When the user is authenticated, a small greeting is displayed at the top right corner of all pages. This authentication enables the user to access the advanced features present in the other pages such as manually setting the sun position, log maintenance and other features.

Figure 5.9: Log In Web Interface Page

5.6.2 Configuration

The configuration page allows the user to manually set the sun position, changing the solar tracker to a manual mode. In order to use this feature the user must be authenticated. It also allows to create another user in the user table of the database.

This page, Figure 5.10, also shows the current info of the solar tracker and allows the user to create graphs in order to see the temperature, zenith and azimuth values in the last days. The graph is dynamically refreshed with the last 100 points of interest (Zenith, Azimuth and Temperature).

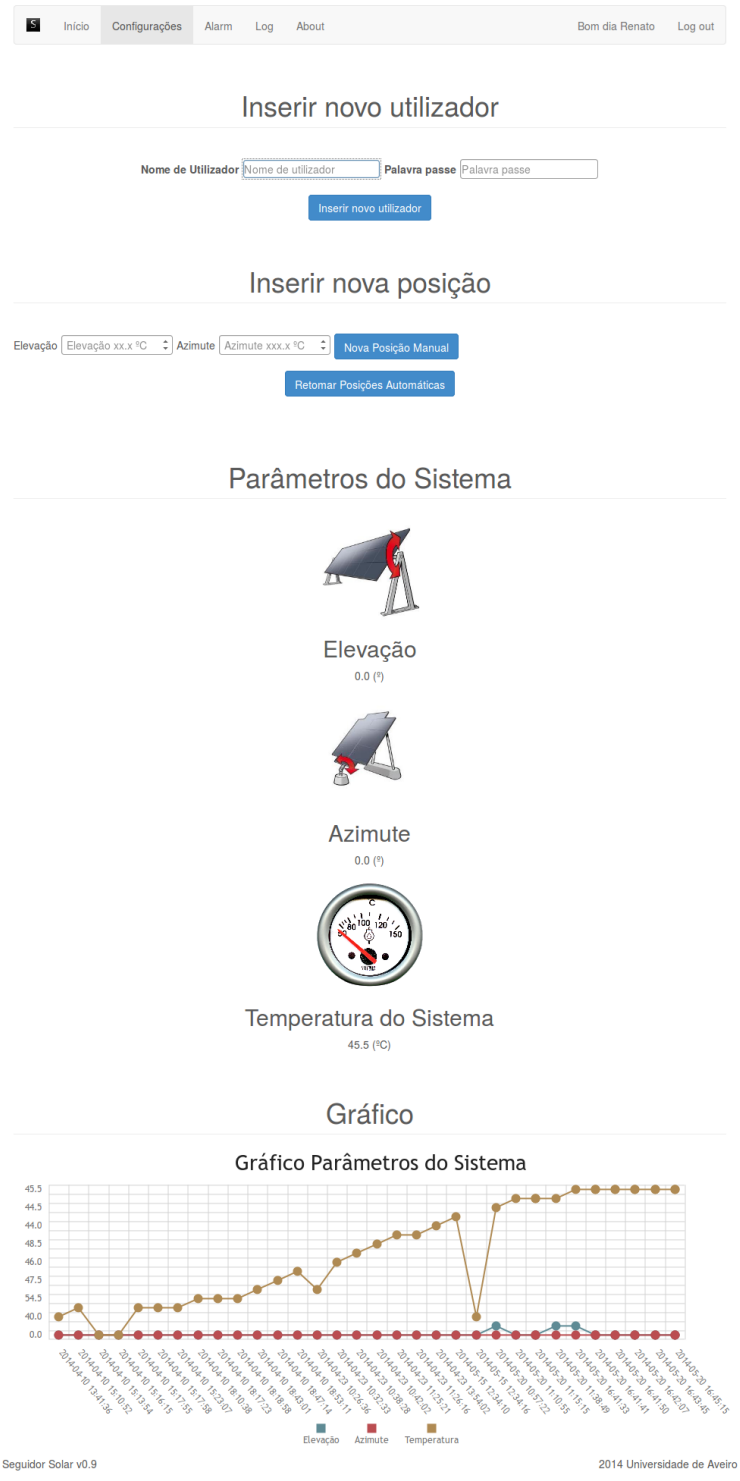


Figure 5.10: Configuration Web Interface Page

5.6.3 Log Viewer

In the log viewer page, Figure 5.11, the user is allowed to check the status log of the solar tracker. He can choose the log entries based on the type of event or by date. Also allows to delete also the log information in the database.

Types of events logged:

- POR - Power On Reset event;
  - Every boot is considered an POR event.
- OVT - Over-temperature event;
  - Logged when transitioning to the *Temperature* state in the state machine.
- HWF - Hardware Failure event;
  - Created when a call to any of the device drivers returns an error.
- SWF - Software Failure event;
  - All SIGTERM, SIGINT (Linux events) and other types of software related errors (MySQL connection errors, etc..).
- PC - Point Calculated Event;
  - When a new set point is calculated (AUTOMATIC MODE).

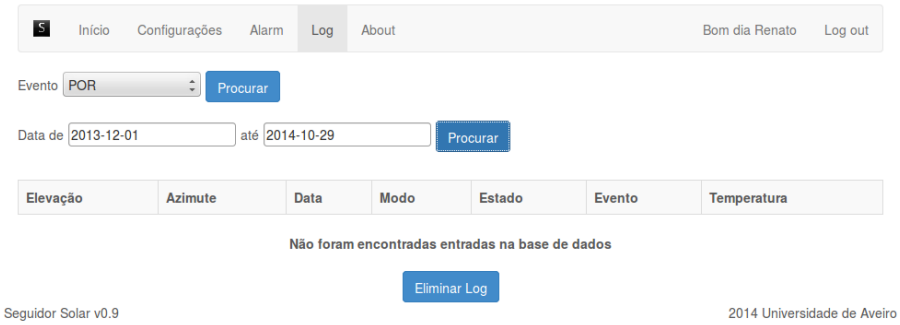


Figure 5.11: Log Web Interface Page

5.6.4 Alarm Configuration

The alarm configuration page, Figure 5.12, allows the user/technician to set alarms in order to be advised of unexpected situations. Allows for alarms about the temperature, Zenith and Azimuth outside of the given range.

Figure 5.12: Alarm Web Interface Page

5.6.5 About

The about page, Figure 5.13, show the information about which version the solar tracker is currently running and other type of information, namely the client information and enterprise name and contacts.

Figure 5.13: About Web Interface Page

Chapter 6

Tests & Results

6.1 Interface Board Prototype

Following Chapters 4 and 5 the final prototype was assembled and ready to test, Figure 6.1.

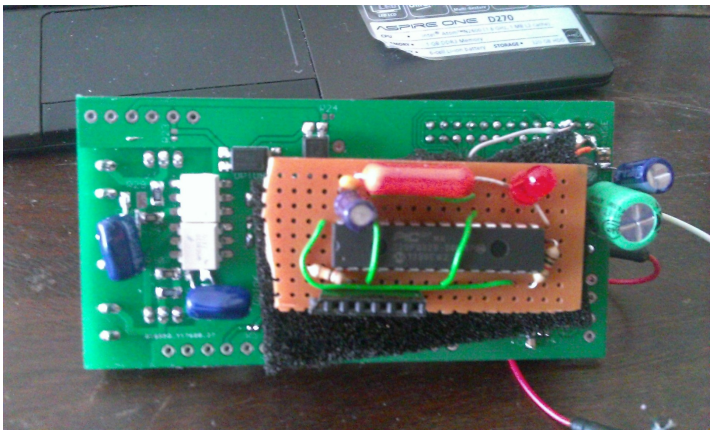


Figure 6.1: Interface Board

To ensure that the prototype created adhered to the requirements specified in Section 3.1, a three step testing was made to ensure that the final result was able to perform as required.

6.2 Hardware Testing

First the hardware was tested to check for malfunctions in the original design and perform the necessary corrections.

6.2.1 DC to DC Converter

The DC to DC converter was tested for various input voltages and output currents, to check the minimum input voltage of operation and the maximum current draw without compromising the output voltage, fixed at 5V DC.

First the DC to DC converter was tested using an 5W 1Ω resistor to check if the hardware current limitation was effectively working. Due to the current limitation the voltage output was $\approx 1.3V$, resulting in a top current of $1.3A$.

Then the voltage output feedback was tested using an 5W 10Ω resistor, in order to draw 500mA from the power source. With this resistor connected to the output, the DC to DC converter was able to maintain the required output mean voltage, 5V, with a ripple voltage of 200mV peak-to-peak.

Then the Raspberry Pi was connected to the power rail in order to ascertain the capability of the power source to properly power the Raspberry Pi. It was tested with the Raspberry Pi connected to an Ethernet cable while running multiple SSH connections, APACHE web server and MySQL. The power supply was able to handle the current needs of the Raspberry Pi with a mean registered temperature of 44.5 degrees Celsius.

When the interface board was fully assembled, the same test was rerun and the power supply was able to properly supply power to all devices in the interface board while maintaining a low voltage output ripple, approximately 200mV peak-to-peak.

6.2.2 MCP23017

The MCP23017 I/O Expander was tested to ascertain the capability of setting/reseting a given pin state or reading a given pin state, depending of the configuration of the pin direction.

In the setting/reseting operation, a particular attention was given to check if any interference existed while changing several pin states at the same time. While monitoring the states all the pins of the port, a command was sent to set the pin state of a given pin and the adjacent pins were monitored to detected any change in state. While running this test several times for all the pins, no interference was detected.

In the read operation the value read from the device was compared to the effective voltages in the inputs pins, after the optical coupling device. No discrepancy was found between the voltage at the input pins and the values read from the device.

It was also tested if the pins connected to the MOC3020 were able to provide the necessary current to enable the TRIAC AC switch. Since a $2K2\Omega$ resistance was used the current draw, for a V_F of 1.2V (Instruments, 1998), was:

$$I_{out} = \frac{5 - V_F}{2200} \approx 1.8mA \tag{6.1}$$

From (Instruments, 1998) we know that, at least, it is necessary a minimum typical current of 15mA. So the resistance used was unable to turn on the TRIAC switch. The resistance were changed to 200Ω which resulted in a I_{out} of 19mA which was sufficient to turn on the TRIAC switch.

The interrupt capability of the device was not tested due to the lack of hardware interrupt capability in the Raspberry Pi.

6.2.3 MCP3002

The requirement in section 3.1.1 stated the a minimum accuracy of 1°. This means that the maximum error in the conversion of the *Zenith* analogue sensor must be (without taking into

account the errors of the conversion that the sensor makes from zenith position to analogue voltage):

$$\begin{aligned} Z_{error} &< \frac{Accurancy * Resolution * AnalogueResolution}{AngleRange} \\ &\equiv Z_{error} < \frac{1 * 1024 * 0.005}{90} \equiv Z_{error} \lessapprox 0.57mV \end{aligned} \tag{6.2}$$

The measured output value of the ADC was measured against the analogue voltage value measured in a oscilloscope to ascertain if the error obtained was outside of the permitted range.

A consistent variation was obtained in the output value of the ADC with an input voltage stable (*i.e.* with a measured variation of less than 10mV) in the range of the ±4 (integer representation of the voltage at the input of the ADC), which results in a variance of, approximately, ±20mV. That means that, from the *Zenith* axis it is possible to reach the desired accuracy.

6.2.4 Optical Coupling

From Section 4.3, the input current to the optical coupler is approximately 2mA. The V_{SAT} will be dependent of the I_C drawn from the interface board. Since in the input optical coupling the I_C is directed at the input of the MCP23017 which as a high input resistance to minimize current consumption.

The I_C for the output optical coupling its trickier to determine. Since its connected directly to a VFD input it is hard to determine its current draw without having the input current specification of the prototype. However, very different outputs resistances were tested, to simulate the input resistance of the VFD. For resistance values bigger than $10K\Omega$ the output voltage in the tests was approximately 23.6V.

6.2.5 TRIAC Switch

To test the TRIAC AC switch an AC fan was connected to the AC output. The AC fan was rated at 90W. To further test the possibility of movement in the two axis simultaneously, another AC device, now a motor rated at 70W, was connected to the other AC output. Given that both AC devices are mainly of the inductive type, which presents problems such as instantaneous current spikes at turn on due to the inductive part, they were the perfect examples to test the VFD enable connections.

Firstly the devices were turn on one at time, then both at the same time and the interface board temperature was monitored to detect if the combined current was affecting the temperature and overheating the board. The temperature remained within the safe operation range, with a median temperature of 50°C.

6.2.6 RTC

The RTC was tested in two parameters:

- - Ability to continue operation while in power failure (powered by the coin cell);

- - Hold information, previously saved to the NVRAM, during power failure;

To test the first situation the current time was set in the appropriate fields of the RTC and it was generated a power failure situation during the course of two weeks. After these two weeks the RTC was connected to the power rail and it was measured the discrepancy between the current time and the time present in the device. Since the resolution of both the Unix/POSIX timestamp and the RTC timestamp is one second, during the course of these two weeks was not possible to detect any discrepancy between the two timestamps.

Another test was made by letting the RTC in the same state during approximately two months where the discrepancy between the two timestamps was merely 3 seconds. Also the voltage in the coin cell was measured and the voltage drop during these two months was negligible. This fact falls in line with the prevision of the 10 year life expectancy of a coin cell connected to the DS1307 at 25°C (Maxim, 2008).

While performing the previous tests, information was stored within the RTC NVRAM, namely the index of the address in the NVRAM was stored at that address resulting in storing the number 8, in 8-bit integer, in the eight address of the NVRAM. In the end of the two months the information stored in the NVRAM was compared to the expected and none discrepancy was found.

6.2.7 Voltage Converters

The *TXS0102* and the *TXB0108* from Texas Instruments are dedicated devices to converting voltage of high frequency signals. However their small package limits the ability to properly solder them into the PCB.

While testing the voltage conversion of the *TXS0102* for the I^2C bus it was observed that it didn't performed correctly. The bus voltage was not being converted from the 3.3V from the Raspberry Pi I^2C controller to the 5V required by the devices in the interface board. While the devices accept 3.3V in the I^2C I/O, it is a suboptimal solution that limits the maximum frequency that they can properly function.

To mitigate this, following the NXP AN10441 application note, we were able to convert the voltages by using a circuit with two MOS-FET to properly connect the two buses.

The *TXB0108* malfunctioned due to the interference of the optical coupling and the input of the MCP23017 port. One of the possible explanation for this fact was due to the functionality of the *TXB0108* that when detects a change in the state, it triggers an one-shot controller effectively diminishing the internal resistance of the I/O port and requesting more current than the current that the optical coupler was able to provide. However when disconnecting the optical coupler from the input of the MCP23017 (due to the fact that was not needed to detect the interrupt since there was no support) the problem did not persist. In the end it was not needed to convert the output of the optical coupler since the PIC32MX220F032B is 5V tolerant in the pins chosen for the inputs that required the TXB0108 (encoder and anemometer).

To convert the SPI bus from the Raspberry Pi to the MCP3002 SPI controller, the TXB0108 performed as expected and was able to properly convert the signal.

6.2.8 Temperature Sensor

To test the values retrieve from the temperature sensor, they were compared against a thermometer from a multimeter and a laser thermometer. The values were inside the expected

range, given the accuracy of $\pm 2°C$ of the device. Since no other function was implemented in the final version of the main application, the other functionalities, such as overtemperature interrupt, were not tested.

6.2.9 External Interrupt Device

The testing of the External Interrupt Device was primarily focused into ascertain the capability of detecting the occurrence of an interruption in the chosen pins.

The chosen pins, *RB7* and *RB10* of the PIC32, were selected due to their 5V tolerant capability which simplified the hardware implementation. These two pins are, by default, input pins to the hardware interrupt controller of the device further simplifying the development of this board.

The hardware was also tested to check if the I^2C was properly connected to the Raspberry Pi I^2C bus, and if the device responded to the commands sent by the Raspberry Pi.

6.3 Software

The software testing was divided into three phases:

- Device-Driver testing;
- State Machine testing;
- Interface testing;

6.3.1 Device Drivers

The device drivers were tested to ascertain the capability of properly controlling the hardware created.

MCP23017

The MCP23017 device driver created was able to control the MCP23017 and implement the functions defined in Section 5.2.2:

- - It is capable of setting/reseting a single pin of the device;
- - Can reset/set several pins of the same port simultaneously, i.e. in the same command.
- - It is capable of reading a single pin if properly configured;
- - Can read the state of the entire port, if configured as input.

The device driver performed as expected and fulfilled the requirements for this device.

MCP3002

The MCP3002 device driver was able to control the device operation and properly convert the voltage signal to the required value.

As required, it allowed to select which channel to convert from and performed as expected.

LM75

The LM75 device driver was capable of correctly convert the temperature from the internal representation to the Celsius format.

DS1307

The DS1307 device driver was capable of setting and getting the current timestamp to and from the device, respectively.

It also allowed to store in the NVRAM values which representation is bigger than one byte, *i.e.* how the information is stored in the RAM, and to retrieve it while maintaining the integrity of the information.

External Interrupt

The testing of the External Interrupt comprehended two different phases:

- Test the counters of the interruptions;
- Test the capability of the I^2C communication to send the counters to the Raspberry Pi.

To simplify the communication with the Raspberry Pi, the counters created were of the byte type having a maximum number of $2^8 - 1$ of interruptions. This represents that, at the 255th interrupt, the counter will overflow and return 0 interrupt occurrences.

However this situation was mitigated recurring to a cyclic function in the main application that at a $100Hz$ rate fetches, as explained before, the information from the counters and clears them. This allows to increase the maximum frequency of the signal that triggers the interrupt to $100 * 255Hz$. Therefore the external interrupt is able to reach a maximum theoretical interrupt frequency of 25500 Hz. However due to the optical coupling, the maximum frequency was approximately $2KHz$.

Board Package

The board package was tested by verifying that the functions created to log the information to MySQL and that the function that implemented the chronological tracking algorithm was returning the expected sun position given a date and the coordinates.

6.3.2 State Machine

The state machine was tested by simulating the necessary conditions to trigger the state change.

The idle state is the default state in which the solar tracker enters when the automatic mode is enabled.

A change in the **Configuration Page** in the **Web interface** triggered, as expected, a transition to the **ManualPos** state.

By simulating the signal of the **Emergency Button** at the Interface Board input, a transition to the **EmergencyPos** state.

A transition to the **AutomaticPos** was generated at a specific interval, which is configurable.

A situation of over-temperature was created to test if the transition to the **Temperature** state occurred.

It was also simulated the existence of wind speed higher that the defined limit, by injecting the necessary signal at the anemometer sensor input. The transition to the **EmergencyPos** state was created as expected.

6.3.3 Command Line Interface

The command line interface was tested by typing the commands to the command line interface and verifying that the executed functions matched the commands given to the application. The gdb linux program, specially designed to debug C/C++ programs was used to check the execution flow of the program while in command line mode.

6.3.4 Web interface

Home

The Home page of the Web Interface was able to correctly show the time at which the main application was started, the time since that moment and the current UTC time and date.

It also correctly showed the *Zenith*, *Azimuth* and *Temperature* values retrieved from the MySQL tables.

The links existent in the Home page were correctly sending the user to the respective page.

Log In

The Log In page was able to initiate a session, which enabled a user to stay logged in while browsing the different pages of the Web Interface, by retrieving the credentials from the MySQL table *users* and compare them to the credentials given by the user.

Configuration

The Configuration page correctly triggered a transition to the **ManualPos** state of the *State Machine* when a manual position was entered and set in the appropriate field.

It also correctly created a new user, if the user requesting the operation was logged in.

The page showed the *Zenith*, *Zenith* and *Temperature* values both in the System Parameters and in the Graph. It automatically refreshed the page without requiring any user action as expected.

Alarm

The Alarm page sets the alarm in the MySQL database. However this feature was not implemented in the main application.

Log

The Log page is able, as required, to retrieve the log information from the database based on user given criteria. This criteria can be by Event or by Date. This action can only be

done if, and only if, the user is logged in. In this situation the user can also remove all log information from the database (perform a maintenance cleanup).

Chapter 7

Conclusions

From Chapter 1 we know that there is currently the need for better and improved solutions to help minimize the damaging effects of global warming. Although with over 100 years of history only now we turned to the solar power as a solution to our energy problem.

In the Chapter 2 we reviewed the major components that nowadays constitute the core of a solar tracker and the solutions implemented to increase the performance of solar trackers.

In Chapter 3 we browsed and review the major market players and chose the requirements which we believe a modern solar tracker controller must satisfy in order to differentiate from the market, shown in 7.1;

	PLC (eg. Siemens)	Expected Controller
Hardware Cost	400-500 €	70€
Programming Tools	400-500 €	Free
Features	Unknown	Free
Maintenance	Local	Local/Remote
Connection	Proprietary Protocol	Ethernet Protocol

Table 7.1: Comparison between solutions

From Table 7.1 we draw the major objectives that this dissertation tried to tackle:

- - **Low Hardware Cost**;
- - Freedom to implement new functionalities;
- - Reliability and Low Maintenance Costs;
- - Ease of use;
- - Use standard and open-source technologies.

In the course of this dissertation we have shown that we have not only decreased the hardware costs by almost 80% but we have done so without compromising any functionality present in the PLC solutions in the market.

This solution is built upon open-source solutions that enable the creation of applications and user interfaces while maintaining development costs down which only benefits the final user.

The web interface created was designed to be easy to use, even to the most unexperienced user, and was kept simple while retaining the necessary functions for the maintenance of the solar tracker.

This solution also differentiate itself from the ones present in the market because, with little or even no increase in the power consumption of the solar tracker controller, we offer an exponential increase in computational power available and at the disposal of the solar tracker. A large number of functionalities that couldn't be implemented in a regular PLC controller can now be implemented.

In Chapter 6 we validated the solution created and tested it to ascertain if it was able to perform as specified in the initial requirements, section 3.1, thus concluding the work in this dissertation.

As Frank Shuman said in the beginning of the century, we know face growing prices in all matters related with energy. We are now rediscovering solar energy and at each step remove the ever so relentless dependency on fossil fuels, thus walking to a brighter future.

7.1 Possible Future Work

The existence of a much bigger computational power in a solar tracker controller opens new possibilities in functionalities. Although the requirements set at the beginning of this dissertation were achieved, during the course of this work, a lot of ideas were disposed because they didn't fit in the initial requirements and as such they were not implemented.

However, it is possible to add value to the solution by implementing the following features:

- Allow various Interface Boards to be controled from a single Raspberry Pi:
  - This feature would allow this solution to be used in large solar power plants that because the panels are relatively closed together there is not the need to independently control each solar tracker;
  - It would also allow energy savings since the Raspberry Pi accounts for as much as half of the power consumed by the controller.
- Implement a remote log server where the controllers would send the log information:
  - This feature would allow technicians to check the log files for any inconsistency and determine whether a visit to the solar tracker location is really necessary.
- From Increase the number of different solar tracker configurations that the controller is able to control:
  - Add active tracking support for multiple type of sun position sensors configurations;

References

Carrier. (2005). *Variable frequency drive.*

C.B., Barros, V., Dokken, D., Mach, K., Mastrandrea, M., Bilir, T., ... (eds.), L. (2014). Summary for policymakers. in: Climate change 2014: Impacts,adaptation, and vulnerability. *Part A: Global and Sectoral Aspects. Contribution of Working Group II to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, 1–32.

EPIA. (2014). *Global market outlook for photovoltaics 2014-2018.*

Gornall, J. (2011). Retrieved from `http://www.thenational.ae/news/uae-news/technology/the-promise-of-solar-power-made-a-century-ago`

Green, M. A., Emery, K., Hishikawa, Y., Warta, W., & Dunlop, E. D. (2012). Solar cell efficiency tables (version 39). *Progress in Photovoltaics: Research and Applications*, *20*(1), 12–20.

Instruments, T. (1998). Retrieved from `http://www.ti.com/lit/ds/symlink/moc3020.pdf`

Maxim. (2008). Retrieved from `http://datasheets.maximintegrated.com/en/ds/DS1307.pdf`

MC34063A. (2010). *1.5 a, step-up/down/inverting switching regulators.* Retrieved from `http://www.onsemi.com/pub_link/Collateral/MC34063A-D.PDF`

McGehee, M. (2014). *Emerging high-efficiency low-cost solar cell technologies* (Tech. Rep.). Stanford University.

Mikkor, A., & Roosmolder, L. (2004). Programmable logic controller in process automation..

Mousazadeh, H., Keyhani, A., Javadi, A., Mobli, H., Abrinia, K., & Sharifi, A. (2009). A review of principle and sun-tracking methods for maximizing solar systems output. *Renewable and Sustainable Energy Reviews*, *13*, 1800–1818.

Reda, I., & Andreas, A. (2008, January). *Solar position algorithm for solar radiation applications* (Tech. Rep.). Golden, Colorado 80401-3393: National Renewable Energy Laboratory.

RS. (2014). *Siemens starter kit.* Retrieved from `http://uk.rs-online.com/web/p/plc-accessories/7704308/`

Siemens. (2014). *How to exploit sunlight toward maximum energy yield?* Retrieved from `http://w3.siemens.com/verticals/mea/en/solar-industry/field-supply/tracking-control/Documents/e20001-a100-t112-x-7600.pdf`

Vishay. (2014). *Sfh-618, optocoupler, phototransistor output, low input current.* Retrieved from `http://www.vishay.com/docs/83673/sfh618a.pdf`

Wikipedia. (2014, January). *Solar tracker.* Retrieved from `http://en.wikipedia.org/wiki/Solar_tracker`

Yokogawa. (2014). *Hxs10-solstation.* Retrieved from `http://www.yokogawa.com/ns/daq/hxs/ns-hxs10_01.htm`